Vers un modèle dynamique du méristème apical caulinaire d'Arabidopsis thaliana

Pierre Barbier de Reuille

UNIVERSITÉ MONTPELLIER II
SCIENCES ET TECHNIQUES DU LANGUEDOC

THÈSE

pour obtenir le grade de

DOCTEUR DE L'UNIVERSITE MONTPELLIER II

Formation Doctorale : Informatique
Ecole Doctorale : Information, Structures, Systèmes

présentée et soutenue publiquement par

Barbier de Reuille Pierre

le 19 Décembre 2005

Vers un modèle dynamique du méristème apical caulinaire d'*Arabidopsis thaliana*

Jury

M. Lebrun Michel	, Président
M. Parcy François	, Rapporteur
M. Tracqui Philippe	, Rapporteur
M. Godin Christophe	, Directeur de thèse
M. Traas Jan	, Co-Directeur de thèse
M. Giavitto Jean-Louis	, Examinateur
M. Konig Jean-Claude	, Examinateur

Remerciements

Tout d'abord, merci à Jan et Christophe pour m'avoir encadré et soutenu pendant cette thèse (et après aussi), et pour m'avoir permis de participer à ce projet.

Merci à Michel Lebrun, François Parcy, Philippe Tracqui, Jean-Louis Giavitto et Jean-Claude König d'avoir accepté de faire partie de mon jury de thèse.

Et merci tout spécialement à Isabelle, avec qui j'ai eu beaucoup de plaisir à travailler et qui a contribuer grandement à ce projet.

Merci aussi à Jean-Louis Giavitto et Olivier Michel pour leur temps et leurs conseils, pour leur accueil dans leur laboratoire.

Merci à Patrick L., Alexis P., Véronique P., Halima M., Nicolas C., Olivier G. et tous ceux que j'ai rencontrés à l'INRA Versailles, pour avoir supporté mes questions (et avoir répondu plusieurs fois à la même question sans même l'avoir fait remarqué), pour leur enthousiasme et pour leur accueil.

Merci à Marianne pour son soutien et ses réponses, même pendant ses vacances.

Merci à Jérôme et Christophe P. pour les nombreuses discussions et tous ces conseils qui m'ont beaucoup aidé pendant ma thèse.

Merci à Yann et Thomas pour leurs conseils en statistique.

Merci à Anne-Laure, Carine, Céline, Chloé, Maryline, Claude-Éric, David, Frédérique, Mickaël G., Michaël L. pour leur soutien, leur bonne humeur, leur accueil, les discussions, les trucs et astuces du thésard ou de l'étudiant à Montpellier.

Merci à tous les membres du laboratoire AMAP pour leur accueil, et merci tout spécialement à Marie-Laure, Nora, Perrine, Sylvie et Yannick pour leur soutien, leur efficacité et leur gentillesse.

Merci à Julien et Daniel pour m'avoir accueilli pendant mes séjours à Versailles et avoir supporté ma conversation pendant ces périodes-là.

Merci à tous mes amis pour leur soutien, leur bonne humeurs, leurs conseils, leur accueil, pour avoir partager mes problèmes : merci à Franck, Manue, Niko, Marsu, Goopil, Gaby, Paula, Michèle, Isabelle, ...

Merci enfin à mes parents, pour m'avoir soutenu et supporté à tout instant dans cette entreprise. Merci aussi à ma grand-mère, mes sœurs Isabelle et Hélène, leurs familles, mes cousins, mes oncles et tantes : Françoise, Jean-Luc, Pierre, Claire, Céline, Gilles, Nadia, Philippe, Fabien, Gaël, Marie-Camille, mon filleul Clément, Julos, Maëlle, Khéna, Olivier, Camille et Louis.

Je remercie enfin tous ceux que j'ai rencontré pendant ma thèse, qui m'ont aidé, soutenu, accueilli, alors que la liste serait trop longue pour figurer ici.

Merci à vous tous sans qui cette thèse aurait été très certainement différente, et assurément moins agréable.

Table des matières

Table des figures

Liste des tableaux

Introduction

La biologie cellulaire bénéficie aujourd'hui de techniques nouvelles accélérant la production de données. Le séquençage complet du génome d'*Arabidopsis thaliana* permet aux chercheurs de trouver et de caractériser les gènes de cette plante bien plus rapidement qu'avant. Les techniques de génie génétique permettent aujourd'hui la création ciblée de mutant, permettant d'observer le phénotype induit par la désactivation ou la sur-expression d'un gène préalablement caractérisé dans une plante vivante. La généralisation de la microscopie confocale et de la microscopie multi-photonique, alliées au génie génétique, permettent d'observer l'évolution dans le temps de l'expression de gènes ciblés.

Devant la quantité de données ainsi générées, il devient nécessaire de développer de nouveaux outils de traitement. D'une part, la quantité de données rend nécessaire une automatisation, au moins partielle, de la saisie. D'autre part, la nature des données est telle que l'extraction de l'information utile est le plus souvent extrêmement complexe et nécessite de nouveaux outils de reconstruction et d'analyse.

Un autre aspect de cette évolution est du à l'utilisation de plus en plus intégrée d'approches génétiques, moléculaires, cellulaires et physiologiques dans l'analyse de ces données. Grâce à ces approches, les avancées de nos connaissances du développement des plantes ont été spectaculaires. La quantité de nouvelles hypothèses est telle qu'il devient difficile d'avoir une vision synthétique du fonctionnement de la plante. Face à ce problème, il est essentiel de développer des outils de modélisation capables d'intégrer les résultats variés et complexes issus des différents champs de la biologie.

Ces avancées permettent notamment une étude bien plus précise d'un organe clef du développement de la partie aérienne de la plante : le méristème apical caulinaire. L'objectif de cette thèse est d'apporter des outils pour la digitalisation, l'analyse et la modélisation de cet organe pour permettre une meilleure exploitation des mesures et des connaissances actuelles. Nous espérons ainsi apporter des éléments de réponse aux questions qui se posent sur la formation des organes latéraux au niveau du méristème et déterminer les questions qui restent à élucider pour la réalisation d'un modèle dynamique de positionnement d'organes.

Ainsi, nous commencerons au chapitre I par présenter les connaissances sur le méristème apical caulinaire sur lesquelles nous nous appuyons pour étudier le positionnement des organes latéraux. Nous verrons notamment les différents modèles d'organisation du méristème, la partie des connaissance sur la génétique et la physiologie du méristème utiles à notre étude et enfin comment un modèle peut déjà être construit à partir de ces informations. Ensuite, nous donnerons un panorama des différents modèles de positionnement des organes latéraux développés depuis un peu moins de deux siècles. Nous partirons des études géométriques permettant de caractériser les différents modes phyllotaxiques

pour aller vers les modèles dynamiques où l'existence même des organes est une propriété émergente en passant par le modèle le plus connu : celui du premier espace libre.

Pour débuter notre approche du problème, nous verrons au chapitre II les outils que nous avons mis en place pour la reconstruction de la géométrie de la surface du méristème et son suivi dans le temps. Nous présenterons d'abord les techniques de traitement d'images utilisées pour retrouver la surface du méristème à partir des images de microscopie confocale. Nous verrons ensuite les outils que nous avons créés pour assister la digitalisation des parois cellulaires. Enfin, nous exposerons une méthode semi-automatique permettant de suivre les divisions cellulaire sur un méristème observé sur plusieurs jours.

Comme cela est rappelé au chapitre I, l'auxine joue un rôle primordial dans le positionnement des organes latéraux. Au chapitre III, nous apporterons des compléments sur la manière dont l'auxine se répartit dans le méristème et nous verrons une proposition d'explication fonctionnelle des flux d'auxines. À cette occasion, nous avons développé un simulateur de méristème pour tester des hypothèses sur le transport d'auxine et observer le résultat sur des méristèmes réels digitalisés. Nous montrerons ainsi comment nous avons représenté mathématiquement et informatiquement le problème biologique du transport d'auxine dans le tissus méristématique.

Après avoir simulé le transport d'auxine dans des méristèmes réels et en avoir observé le résultat, nous donnerons au chapitre IV la méthodologie que nous avons suivi pour construire un modèle dynamique de positionnement d'organes au niveau du méristème. Nous donnerons d'abord les divers choix de modélisation et leurs expressions mathématiques et informatiques. Puis, nous analyserons les résultats obtenus avec le modèle ainsi construit. Enfin, ce projet a été l'occasion de tester l'utilisation d'un langage dédié à ce type de modélisation. Nous donnerons donc les avantages et les inconvénients du langage que nous avons choisi pour le modèle que nous avons élaboré.

Enfin, au chapitre V, nous verrons comment tous les outils informatiques développés au cours de cette thèse ont été assemblés dans deux logiciels, dont une plate-forme d'étude du méristème. Ce type de plate-forme a pour objectif de permettre l'intégration d'outils développés par des équipes diverses, mais aussi de mettre à disposition des biologistes autant que des informaticiens les outils créés et assemblés pour les besoins de cette thèse. Pour cela, nous avons utilisé des techniques modernes de génie logiciel, que nous tenterons d'évaluer dans ce cadre.

Chapitre I

Biologie et modélisation du méristème

La première observation de ce que nous appelons aujourd'hui le méristème apical caulinaire* (MAC) date de 1760 les travaux de Wolff (1896)[1](cité de Tooke et Battey, 2003). Il a découvert, à une époque où les microscopes étaient relativement peu évolués, le site de croissance de la plante, le "*punctum vegetationis*" (Figure I.1), lieu de formation des feuilles.

Mais ça n'est que bien plus tard que l'étude de cet organe et du positionnement des feuilles a vraiment pris son essor, avec principalement les travaux de Schimper en 1830, de Braun en 1831, des frères Bravais en 1837 et de Schleiden en 1842.

La conception du méristème d'aujourd'hui n'est pas différente de celle de Wolff, mais plus précise, plus complexe et plus étayée par 250 ans d'observations, d'expériences et de modèles. Nous allons donc décrire dans un premier temps les connaissances biologiques actuelles sur le MAC et dans un deuxième temps, les différents modèles qui ont été développés au cours de ces deux derniers siècles (pour revue : Tooke et Battey, 2003).

I.1 Biologie du méristème

Contrairement à l'animal, la plante produit ses organes tout au long de sa vie en des zones bien définies : les méristèmes. Les méristèmes contiennent des cellules souches qui, par homologie avec la définition utilisée pour les cellules animales, sont des cellules indifférenciées présentes dans certains tissus adultes, capables de s'auto-reproduire et de donner naissance à différents types cellulaires (pour revue : Weigel et Jürgens, 2002). Les méristèmes ont ainsi la capacité de s'auto-maintenir et de créer de nouveaux organes tout au long de la vie de la plante. On distingue deux catégories de méristèmes : les méristèmes primaires qui assurent la croissance longitudinale de la plante et les méristèmes secondaires qui assurent sa croissance radiale (Figure I.2).

Parmi les méristèmes, le MAC revêt une importance particulière. D'une part il est présent dès l'embryogénèse (Long *et al.*, 1996) et jusqu'à la mort de la plante, comme le méristème apical racinaire. D'autre part, il est responsable de la mise en place de

[1]la première édition a été imprimé en 1759, 1896 est la date de la troisième édition en allemand

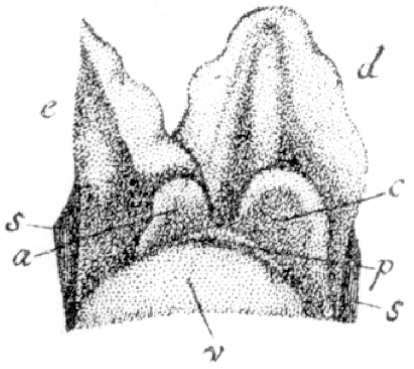


FIG. I.1: **Le "*punctum vegetationis*".** Diagramme de Casper Wolff (Wolff, 1896) qu'il décrit par : "Le sommet a été exposé et toutes les feuilles de la face avant enlevées pour qu'il soit possible de voir le site végétatif. Les feuilles de la face arrière n'ont pas été enlevées pour montrer leur attachement à la surface du végétal. (v) La surface convexe, humide et translucide du végétal. (p) La première feuille qui apparaît, avec sa surface interne concave adjacente à la surface du végétal. La consistance de cette feuille est à peine plus substantielle qu'un liquide visqueux. (a) Une autre feuille, plus large et plus substantielle que la précédente. (c) Une feuille qui a déjà développé un bord. (e) Demi-feuille. (d) Feuille complète." (Figure reproduite de la revue : Tooke et Battey, 2003)

l'ensemble de la partie aérienne de la plante et du positionnement des organes latéraux. Enfin, le méristème peut être étudié à toutes les échelles de la plante. À l'échelle de la plante entière, la phyllotaxie* est le résultat direct du fonctionnement du MAC (mais aussi de l'allongement des entre-nœuds). Au niveau de l'organe, une organisation est déjà très visible (voir section I.1.1.1). À l'échelle tissulaire, les études histologiques et cytologiques révèlent des propriétés fonctionnelles des différentes zones du méristème (voir sections I.1.1.2 et I.1.1.3). Enfin, à l'échelle sub-cellulaire, les études bio-chimiques et génétiques nous permettent d'entrevoir les réseaux génétiques et la signalisation inter-cellulaire en jeu dans le fonctionnement du méristème (voir section I.1.2).

Étant donné la grande variabilité existant entre les plantes, nous nous limiterons aux cas des angiospermes et, plus spécifiquement, à *Arabidopsis thaliana*.

I.1.1 Structure du méristème

I.1.1.1 Structure de l'organe

La structure macroscopique du MAC donne une bonne idée de son fonctionnement. Tout au long de sa vie il produit des unités séquentielles appelées phytomères presque sans interruption. Chaque phytomère est composé d'un organe latéral (un nœud) et d'une portion de tige appelée entre-nœud. Comme chez tous les angiospermes*, le MAC d'*Arabidopsis thaliana* passe par différentes phases au cours du développement de la plante caractérisées par un changement de l'identité des organes latéraux produits. Le MAC végétatif produit des feuilles présentant des méristèmes axillaires à leur aisselles. Après la transition florale, le MAC est appelé méristème d'inflorescence et produit des feuilles caulinaires* avec des méristèmes axillaires à leurs aisselles (qui produiront des inflorescences

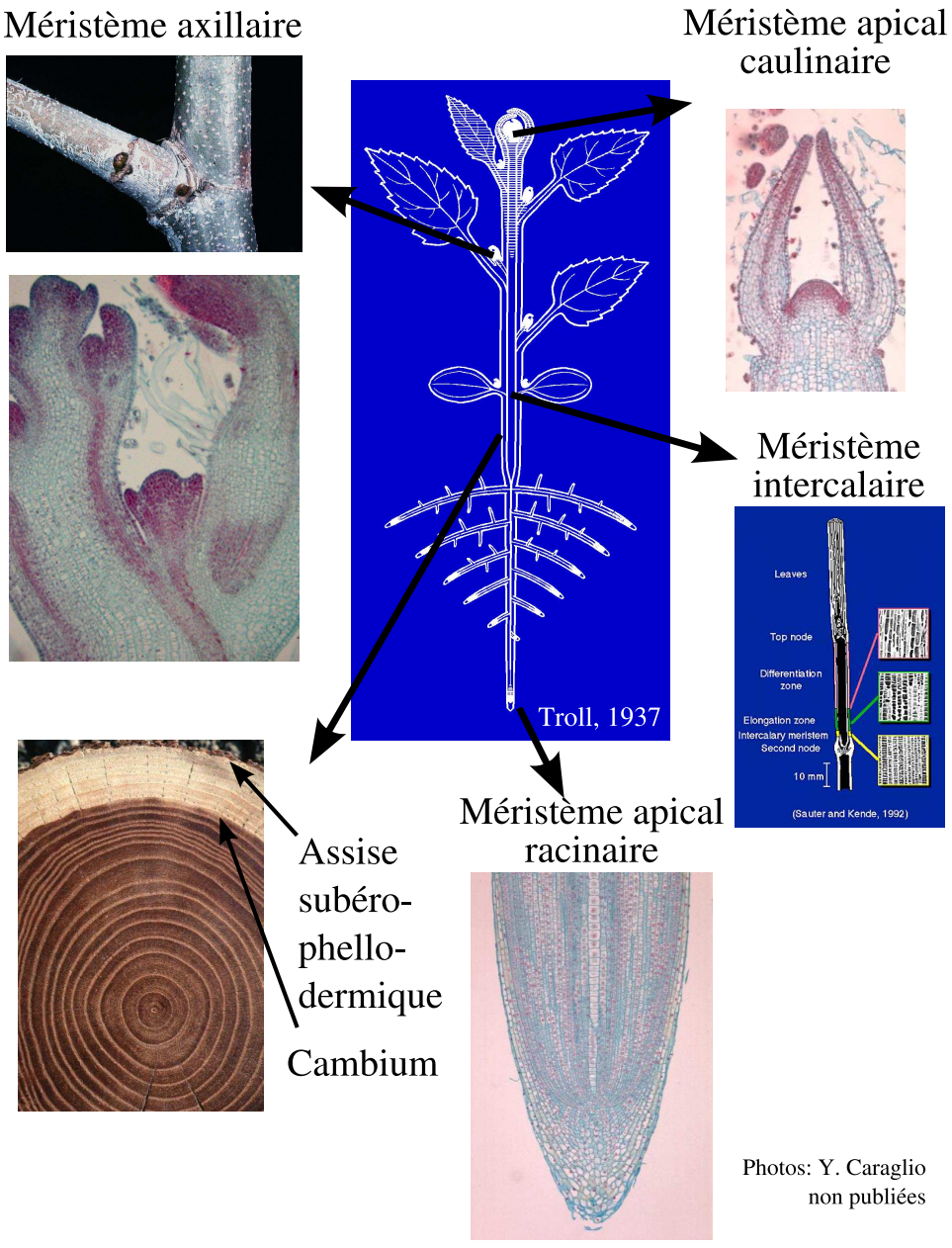


FIG. I.2: **Position des différents méristèmes chez les végétaux supérieurs.** Les méristèmes apicaux, axillaires et intercalaires sont appelés méristèmes primaires et assurent la croissance longitudinale de la plante. Le cambium et l'assise subéro-phellodermique assurent quant à eux la croissance radiale et sont appelés méristèmes secondaires.

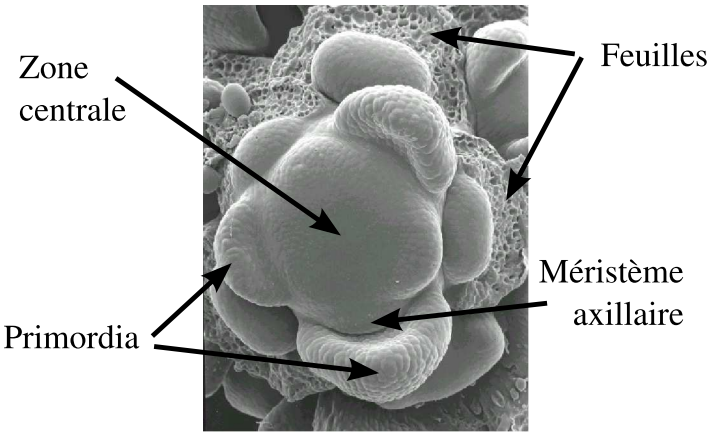


FIG. I.3: **Vue d'un MAC au microscope électronique.** Le MAC a une structure apparente très visible. En son centre, il existe une zone relativement plane constituée de petites cellules. Cette zone centrale est entourée de bosses qui formeront les futures feuilles et organes et sont appelées primordia. Enfin, à l'extérieur, les marques des feuilles qui ont été enlevées sont visibles car les jeunes feuilles recouvrent normalement le méristème pour le protéger.
Image : MAC végétatif d'*Antirrhinum*, Traas et Doonan (2001)

axillaires), puis des méristème floraux sans bractees⋆. Chez *Arabidopsis*, les MAC sont indéterminés et capables de produire de nouvelles structures tout au long de la vie de la plante.

Une observation macroscopique d'un MAC (Figure I.3) montre une zone centrale plane composée de petites cellules, probablement responsables du maintien de l'identité méristématique. Autour, on peut voir les primordia, qui formeront les organes latéraux et qui apparaissent à une distance à peu près constante du centre, selon une disposition qui reflète la phyllotaxie⋆ observable à l'échelle macroscopique.

Pour aller plus avant dans l'étude de la structure du méristème, il est nécessaire de recourir à une étude histologique⋆ et cytologique⋆.

I.1.1.2 Assises cellulaires

L'observation de coupes longitudinales révèle aisément une structure en assises du MAC (Figure I.4A). L'assise externe, appelée *tunica*, entoure l'assise interne constituant le *corpus*. Cette organisation en assises découle d'une orientation quasi-strictement anticlinale⋆ des plans de division des cellules de la *tunica*. Chez les dicotylédones⋆, la *tunica* est en général elle-même composée de deux assises appelées L1 et L2 (pour revue : Steeves et Sussex, 1989). Les plans de division des cellules du *corpus* n'ont pas, globalement, d'orientation préférentielle. Aussi, cette assise n'est pas stratifiée comme la *tunica*.

Des études basées sur l'analyse de chimères⋆ périclines⋆ (voir Figure I.5) permettent de déterminer la contribution des assises cellulaires à la formation des différents tissus composant les organes de la plante. En général, chez les dicotylédones⋆, les cellules descendant de la L1 forment les tissus épidermiques. Les cellules qui forment les tissus sous-épidermiques

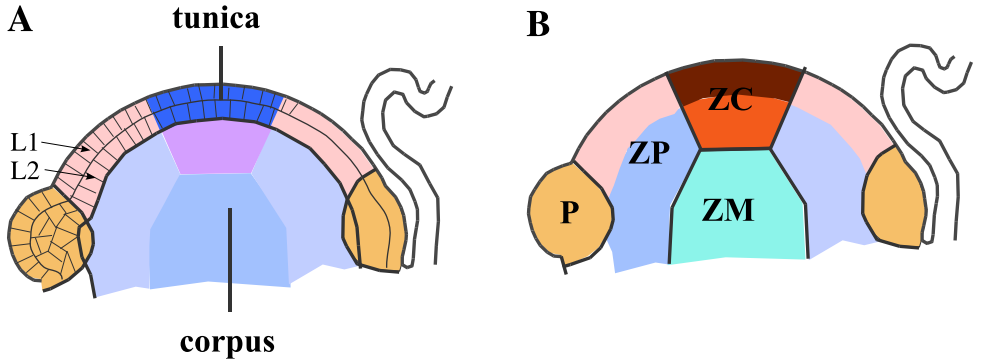


FIG. I.4: **Modèle d'organisation du méristème apical caulinaire des angiospermes.** (**A**) : Modèle d'organisation en assises cellulaires. La *tunica* est composée de deux ou trois assises L1, L2 et, éventuellement, L3. Le *corpus* est composé de cellules dont l'organisation n'est pas aussi évidente. (**B**) : Modèle d'organisation en zones concentriques. ZC : zone centrale, ZP : zone périphérique, ZM : zone médullaire⋆, P : primordium. (A) montre aussi la superposition des deux modèles.
Adapté de Traas et Doonan (2001).

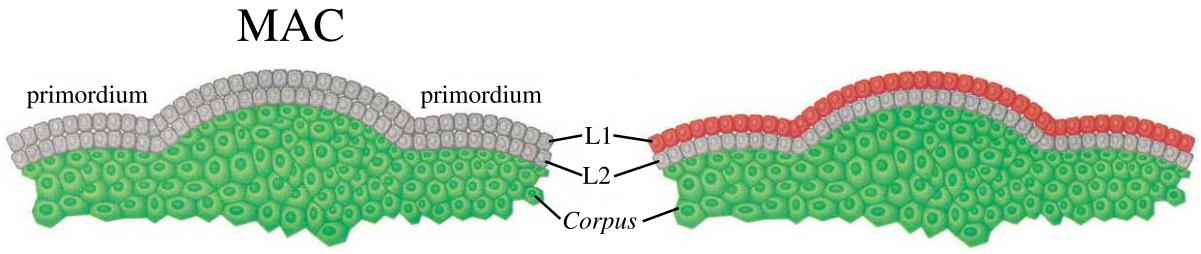


FIG. I.5: **Les chimeres périclines.** Les plantes chimériques⋆ sont constituées de cellules qui différent pour un ou plusieurs marqueurs. Dans les chimères⋆ périclines⋆, chaque assise se voit associer un marqueur différent. Ainsi, chaque cellule passant par une des assises active ce marqueur. Ceci permet de tracer le "passage" des cellules dans les différentes assises (i.e. les cellules dont un ancêtre au moins était dans l'assise marquée). (L1, L2 et *corpus* sont les différentes assises cellulaires du méristème : cf. figure I.4)
MAC : méristème apical caulinaire⋆.
Adapté de Wu *et al.* (2002).

et les gamètes dans les fleurs sont issues de la L2. Enfin, les tissus internes des organes latéraux et de la tige sont formés de cellules venant du *corpus*. Toutefois, le devenir d'un lignage ne dépend pas de ses origines mais de la dernière position qu'il occupe. Ainsi, si une cellule change d'assise, son évolution ne dépend que de la dernière assise dans laquelle elle se trouve (Meyerowitz, 1996).

I.1.1.3 Zones concentriques

Une étude cytologique⋆ plus fine permet de subdiviser le MAC en zones concentriques (voir Figure I.4B, pour revue : Nougarède, 1967; Steeves et Sussex, 1989; Lyndon, 1998). Ces différents domaines ont été initialement décrits par Foster en 1928 pour le MAC du Gingko, et ce modèle d'organisation en zones concentriques semble être une caractéristique de toutes les espèces de plantes supérieures terrestres (Steeves et Sussex, 1989). Trois domaines peuvent être définis. La zone centrale (ZC) est située au sommet du méristème. Elle est entourée par la zone périphérique (ZP), lieu de l'initiation des primordia. La zone médullaire (ZM) constitue pour sa part la partie interne sous-jacente à la ZC et à la ZP, et est caractérisée par des files cellulaires verticales.

Dans beaucoup d'espèces, la ZC est caractérisée par des cellules plus grandes avec un gros noyaux et davantage vacuolisées que la ZP. Ces cellules sont colorées moins intensément par certains colorants cytologiques⋆, suggérant une activité métabolique réduite. Toutefois, ces caractéristiques ne sont pas apparentes partout, et notamment pas chez *Arabidopsis* (Laufs *et al.*, 1998). Une autre caractéristique permettant de différencier la ZC de la ZP est la durée du cycle cellulaire. Les cellules du sommet de l'apex se divisent en général plus lentement que les cellules situées sur les flancs de l'apex (Nougarède, 1967; Steeves et Sussex, 1989; Lyndon, 1998). Chez *Arabidopsis*, la ZC a un diamètre de 4 à 6 cellules dans un méristème d'inflorescence (Laufs *et al.*, 1998). Par contre, la taille de la ZC n'a pas pu être déterminée dans le méristème végétatif, même si des différences ont pu être détectées entre la périphérie et le centre du méristème.

Lors de l'apparition d'un primordium dans la ZP, le taux de division cellulaire et la vitesse de croissance au site d'émergence augmentent significativement (Lyndon, 1970; Laufs *et al.*, 1998; Kwiatkowska et Dumais, 2003; Grandjean *et al.*, 2004; Reddy *et al.*, 2004). Des divisions périclines⋆ sont observées dans la L2 et une croissance polarisée d'une partie des cellules selon l'axe d'émergence de l'organe marque le développement de la nouvelle structure.

I.1.2 Étude biochimique et génétique du méristème

Les deux modèles cyto⋆-/histologiques⋆ d'organisations du MAC (en assise et en zones) ont une correspondance fonctionnelle. En effet, en ce qui concerne l'organisation en zones concentriques, il est maintenant communément admis que la ZC est composée de cellules souches pluripotentes⋆ qui alimentent l'ensemble de la structure méristématique et permettent l'auto-maintien du méristème. La ZP est le lieu d'initiation des organes et la ZM génère les parties internes de la tige. Pour l'organisation en assises, il est, entre autre, généralement admis que le positionnement des primordia est du à des processus ayant lieu principalement dans la L1 (Reinhardt *et al.*, 2003a,b; Vernoux *et al.*, 2000).

L'existence de ces modèles d'organisation a pu être validée et complétée par des approches de génétique moléculaire dont nous allons donner maintenant des éléments à travers deux aspects : l'auto-maintien du méristème et l'initiation des organes.

I.1.2.1 Auto-maintien du méristème apical caulinaire

Comme nous l'avons déjà évoqué, une fois mis en place, le MAC conserve sa structure jusqu'à la mort de la plante. De plus, si un événement extérieur provoque la disparition du MAC, un (ou plusieurs) nouveau MAC est formé qui reprend la fonction du précédent.

Initiation et maintien de l'identité méristématique. Des gènes clefs de l'identité méristématique se trouvent être des gènes à homéoboîte⋆. Les gènes à homéoboîte⋆ codent pour des facteurs de transcription⋆ et sont des régulateurs essentiels du développement des eucaryotes multicellulaires (pour revue : Kappen, 2000).

Chez *Arabidopsis thaliana*, un premier gène a été identifié lors de la caractérisation du mutant *shoot meristemless*[2] (*stm*), qui implique un gène de la famille des gènes à homéoboîte⋆ *KNOTTED* (*KNOX*, pour "*KNOTTED*-like homeobox"). Les allèles⋆ forts de ce mutant sont incapables de développer un méristème fonctionnel, suggérant que le gène *STM* soit impliqué dans la formation et la maintenance du MAC. Cependant, ces mutants sont capables d'initier sporadiquement des organes et de produire des méristèmes ectopiques⋆ après la germination (Barton et Poethig, 1993; Endrizzi *et al.*, 1996), suggérant l'existence d'une activité méristématique résiduelle. Dans la plante adulte, le gène *STM* est exprimé dans tout le MAC sauf dans les primordia où il est réprimé dès le recrutement des cellules fondatrices des organes. Dans l'embryon globulaire, l'ARN messager (ARNm) *STM* est déjà présent dans une à deux cellules au sommet, suggérant une mise en place du MAC très précoce, bien avant qu'il ne soit morphologiquement visible.

L'existence d'une activité méristématique résiduelle dans le mutant *stm* suggère l'existence de gènes partiellement redondants. Parmi les gènes *KNOX*⋆ d'*Arabidopsis*, trois autres (*KNAT1*, *KNAT2* et *KNAT6*) ont été impliqués dans la fonction méristématique. Le premier, *KNAT1*, est exprimé dans les MAC végétatifs et d'inflorescences dans un domaine qui recouvre en grande partie celui de *STM* (Lincoln *et al.*, 1994). Une sur-expression du gène *KNAT1* sous contrôle du promoteur constitutif de la mosaïque du chou-fleur (*35S*), induit la formation de feuilles lobées et de méristèmes ectopiques⋆ sur la face supérieure des feuilles (Lincoln *et al.*, 1994; Chuck *et al.*, 1996). *KNAT1* pourrait donc jouer un rôle similaire à celui de *STM* dans la fonction méristématique. *KNAT2* a un domaine d'expression plus restreint. Il n'est pas actif dans le MAC d'inflorescence, mais est présent dans le MAC floral et dans une partie des carpelles. Par ailleurs, sa sur-expression, si elle produit des feuilles lobées, ne suffit pas à créer des méristèmes ectopiques (Pautot *et al.*, 2001). La séquence de *KNAT6* est très proche de celle de *KNAT2*, indiquant que ces deux gènes pourraient intervenir de manière redondante, mais la fonction de *KNAT6* dans le MAC reste à définir.

[2]par convention, nous noterons en majuscules italiques le nom des gènes (*STM*), en majuscules droites le nom des protéines (STM) et en minuscules italiques le nom des mutants perte de fonction (*stm*)

Régulation de la zone centrale. Un autre gène à homéoboîte⋆ a été identifié grâce à la caractérisation du mutant *wuschel* (*wus*) d'*Arabidopsis* (Laux *et al.*, 1996; Mayer *et al.*, 1998). Les mutants *wus* sont incapables de maintenir un MAC fonctionnel. Mais à la différence des mutants *stm*, le MAC est mis en place puis cesse de fonctionner et se désorganise progressivement après avoir formé quelques organes (Laux *et al.*, 1996). Ils réinitient alors des méristèmes ectopiques⋆ qui présentent des défauts similaires et cessent de fonctionner prématurément. Dans la plante adulte, le gène *WUSCHEL* est exprimé dans un petit groupe de cellules au centre du MAC sous les trois couches cellulaires les plus externes, ce qui correspond à la partie basale de la ZC ou à une partie sous-jacente à la ZC. Le gène *WUS* permettrait de spécifier l'identité des cellules souches de manière non cellule-autonome⋆ et jouerait le rôle de centre organisateur du méristème (Mayer *et al.*, 1998).

Un allèle⋆ fort de la mutation *stm* est épistatique⋆ sur la mutation *wus*, suggérant que *STM* pourrait agir en amont de *WUS*. Cependant, l'expression des gènes *STM* et *WUS* est correctement initiée respectivement dans les mutants *wus* et *stm* (Mayer *et al.*, 1998), mais plus tard ils disparaissent du MAC adulte. Ainsi, les profils d'expression des deux gènes ainsi que l'indépendance de l'initiation de leur expression indique qu'ils agissent sur des niveaux de régulation de la fonction méristématique différents (Mayer *et al.*, 1998). L'épistasie⋆ de *stm* sur *wus* pourrait alors signifier que la mutation de *stm* affecte le développement embryonnaire avant la mutation *wus*.

Les gènes *CLAVATA 1*, *2* et *3* (*CLV*) sont impliqués dans le contrôle de la taille du méristème (Clark *et al.*, 1993, 1995; Kayes et Clark, 1998). Plus spécifiquement, les mutants *clv1*, *clv2* et *clv3* présentent un MAC anormalement étendu (jusqu'à 1mm de diamètre). Cette augmentation de la taille, visible dès l'embryogénèse, se fait progressivement et est accompagnée d'une surgénération d'organes latéraux (feuilles, organes floraux). Les analyses génétiques suggèrent que ces trois gènes fonctionnent dans une même voie de signalisation. Les mutants *clv1* et *clv3* ont un phénotype identique, indifférenciable du double mutant *clv1 clv3* (Kayes et Clark, 1998). Par contre, le phénotype de *clv2* est moins marqué, les allèles⋆ forts de *clv2* ayant un phénotype comparable aux allèles faibles de *clv1* ou *clv3*. De plus, les allèles forts de *clv1* et *clv3* sont épistatiques⋆ sur les allèles faibles de *clv2*, ce qui est une confirmation de l'implication de ces trois gènes dans le même processus de régulation de la fonction méristématique. Toutefois, alors que *CLV1* et *CLV3* semblent limités au MAC lui-même, *CLV2* semble jouer un rôle hors du méristème.

Une étude détaillée du mutant *clv3* a permis de conclure que l'augmentation de la taille du méristème se fait par augmentation du nombre de cellules dans la ZC (Laufs *et al.*, 1998). La caractérisation moléculaire des gènes *CLV* a permis de confirmer son implication dans le contrôle de la ZC en limitant la taille de la population de cellules souches dans le MAC.

Des études sur des chimères⋆ périclines⋆, obtenues à partir de la réversion dans une assise du MAC d'un allèle⋆ de *clv3* instable, indiquent que la présence de *CLV3* dans une seule assise du MAC est suffisante pour assurer un fonctionnement méristématique normal (Fletcher *et al.*, 1999). *CLV3* fonctionne donc de manière non cellule-autonome⋆. L'analyse biochimique des protéines CLV1, CLV2 et CLV3 et de leurs interactions indique que CLV3 serait le ligand du complexe formé par CLV1 et CLV2. Le gène *CLV3* n'est actif qu'au sommet du méristème, très probablement dans la ZC, alors que *CLV1* et *CLV2* sont

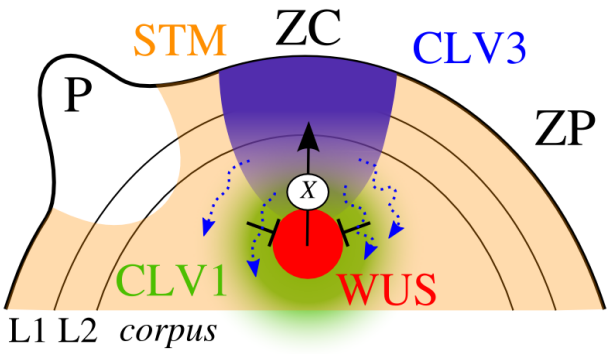


FIG. I.6: **Modèle de maintien du MAC.** Ce schéma montre les zones d'expression des gènes connus pour être impliqués dans l'auto-maintien du MAC. Les flèches en pointillé indiquent la diffusion de la protéine CLV3, qui peut se combiner à CLV1 pour réprimer *WUS*. À son tour, la protéine WUS va activer, probablement par l'intermédiaire d'un facteur X, le gène *CLV3*. Quant à *STM*, on sait qu'il est exprimé dans tout le MAC, sauf à l'endroit des primordia, et qu'il est nécessaire au maintien de la fonction méristématique, mais on ne sait pas comment il s'inscrit dans ce réseau génétique.

actifs dans les couches internes du MAC (voir figure I.6). Le complexe formé par CLV1 et CLV2 comporte un récepteur kinase (CLV1), une partie membranaire et un récepteur à ligand ciblé par CLV3. Vraisemblablement, ce complexe se fixe sur la membrane cellulaire, le récepteur de CLV3 à l'extérieur de la cellule et le récepteur kinase⋆ à l'intérieur. Ainsi, CLV3 agit depuis l'extérieur de la cellule pour catalyser, par l'intermédiaire de CLV1 et CLV2, des réactions internes (Trotochaud *et al.*, 1999; Jeong *et al.*, 1999).

Modèle d'auto-maintien du MAC. Une cible importante de la voie CLV est la protéine à homéodomaine⋆ WUS. Divers éléments suggèrent que *WUS* agit en amont des gènes *CLV*. D'une part, *wus* est épistatique⋆ sur les trois mutations *clv*. D'autre part l'expression de *WUS* régule positivement l'expression de *CLV3*. Mais d'autres éléments suggèrent que c'est *CLV3* qui agit en amont de *WUS*, en particulier l'expression de *CLV3* régule négativement celle de *WUS*. Ces résultats semblent montrer que la fonction d'auto-maintien du MAC est le résultat d'une boucle de rétroaction faisant intervenir *CLV3* et *WUS* (Figure I.6; Schoof *et al.*, 2000; Brand *et al.*, 2000).

Il est par ailleurs certain que *STM* joue un rôle dans l'auto-maintien du méristème, mais il n'a pas encore été déterminé avec précision.

I.1.2.2 Initiation des organes

Une autre fonction clef du MAC est l'initiation des organes, appelés à ce stade "primordium". La formation d'un organe implique la perte de la fonction méristématique d'un groupe de cellules pour acquérir la fonction "organe" (le recrutement), la mise en place de frontières qui isolent le primordium du reste du méristème, l'émergence du primordium et l'identification de l'organe (i.e. feuille ou fleur).

Recrutement des cellules fondatrices. Le recrutement des cellules fondatrices implique à la fois la perte de l'identité méristématique et l'acquisition de l'identité "organe".

Comme nous l'avons déjà vu, dans beaucoup d'espèces, des gènes *KNOX*⋆ tels que *STM* ou *KNAT1* sont normalement réprimés dans les cellules fondatrices des organes (section I.1.2.1; Byrne *et al.*, 2000). Ces répressions requièrent la présence de facteurs de transcription de la famille MYB, codés par les gènes orthologues⋆ *ROUGH SHEATH 2* (*RS2*; Schneeberger *et al.*, 1998; Timmermans *et al.*, 1999; Tsiantis *et al.*, 1999), *PHANTASTICA* (*PHAN*; Waites *et al.*, 1998) et *ASYMMETRIC LEAVES 1* (*AS1*; Byrne *et al.*, 2000), respectivement chez le maïs, *Antirrhinum* et *Arabidopsis* (pour revue : Tsiantis, 2001). Les gènes *PHAN* et *AS1* sont exprimés dans les cellules fondatrices, alors que le gène *RS2* n'est exprimé que légèrement après le recrutement, juste avant l'émergence du primordium. Une mutation "perte de fonction" dans un de ces gènes a un phénotype foliaire proche du phénotype associé à une sur-expression des gènes *KNOX*⋆ (Timmermans *et al.*, 1999; Tsiantis *et al.*, 1999; Byrne *et al.*, 2000; Ori *et al.*, 2000; Semiarti *et al.*, 2001). Chez le mutant *as1* d'*Arabidopsis*, les gènes *KNAT1*, *KNAT2* et *KNAT6* sont exprimés ectopiquement⋆ chez les feuilles en développement (Byrne *et al.*, 2000; Ori *et al.*, 2000; Semiarti *et al.*, 2001), suggérant que *AS1* pourrait réprimer leur expression dans les primordia.

Toutefois, les mutations *as1*, *rs2* et *phan* n'affectent pas l'expression des gènes *KNOX*⋆ dès le recrutement des cellules du primordium. L'extinction de l'expression des gènes *KNOX*⋆ se fait normalement dans ces mutants, ce n'est qu'après qu'ils sont réactivés (Timmermans *et al.*, 1999; Tsiantis *et al.*, 1999; Ori *et al.*, 2000). Il semble donc que *AS1* soit important pour maintenir l'identité "organe" d'un groupe de cellules mais qu'il ne soit pas responsable de la perte de l'identité méristématique.

Établissement des frontières des primordia. Alors que les cellules fondatrices des organes sont recrutées, les frontières sont mises en place. L'importance de ces frontières a été mise en évidence par l'analyse de mutants dans des facteurs de transcription putatifs de la famille NAC (NAm, Cup-shaped cotyledon).

Une mutation dans l'un ou l'autre des gènes codant pour les facteurs de transcription de la famille NAC, CUP-SHAPED COTYLEDON1 (CUC1) ou CUC2 ne provoque qu'un phénotype léger où la plante se développe presque normalement. Mais une mutation dans les deux gènes simultanément induit la fusion des cotylédons⋆ pratiquement sur toute leur hauteur, ce qui leur donne cette forme caractéristique en coupe (Figure I.7; Aida *et al.*, 1997, 1999).

Pendant l'embryogénèse, les gènes *CUC* sont exprimés dans toute la zone entre les deux cotylédons⋆, puis sont progressivement exclus de la zone centrale du méristème, ne s'exprimant alors que dans les frontières entre les cotylédons et le méristème, et entre les cotylédons eux-mêmes (Aida *et al.*, 1999; Ishida *et al.*, 2000; Takada *et al.*, 2001). Ainsi, il a été montré que les gènes *CUC1*, *CUC2* et *CUC3* sont impliqués dans la séparation des cotylédons (Vroemen *et al.*, 2003). Ainsi, les mutants simples *cuc1*, *cuc2* ou *cuc3* ne présentent pas de phénotype marqué, alors que le double mutant *cuc1 cuc2* présente une fusion des cotylédon, avec toutefois un plan de symétrie visible permettant d'identifier les deux cotylédons fusionnés. Enfin, le phénotype du triple mutant *cuc1 cuc2 cuc3* montre des cotylédons en forme de coupe parfaite sans plan de symétrie apparent. Le double

FIG. I.7: **Phénotype d'un mutant *cup-shaped cotyledon* d'*Arabidopsis thaliana*.** Le mutant *cuc* est un double mutant dans les gènes redondants *CUC1* et *CUC2*. Cette figure montre des plantules de 10 jours en culture *in vitro*. (A) : phénotype du mutant simple *cuc1* ou *cuc2*. Il est indiscernable de la plante sauvage. (B) : phénotype du double mutant *cuc1 cuc2*. Les cotylédons* sont fusionnés des deux côtés, formant un cotylédon unique en forme de coupe et le MAC est absent. (Vernoux, 2002)

mutant *cuc1 cuc2* Par ailleurs, le double mutant *cuc1 cuc2* ne peut pas former de MAC pendant l'embryogénèse car il n'exprime pas le gène *STM* (Aida *et al.*, 1999). De plus, une sur-expression du gène *CUC1* sous contrôle du promoteur constitutif *35S* induit la formation de méristèmes ectopiques* principalement sur la face adaxiale* des cotylédons (Takada *et al.*, 2001). Ces deux résultats semblent indiquer que les gènes *CUC1* et *CUC2* activent de manière redondante le gène *STM* pendant l'embryogénèse. Symétriquement, *STM* semble réguler indirectement l'expression de *CUC2* (Aida *et al.*, 1999).

Dans les méristèmes d'inflorescences et les méristèmes floraux, *CUC1* et *CUC2* sont également exprimés à la frontière des organes (Ishida *et al.*, 2000; Takada *et al.*, 2001). Ces profils d'expression suggèrent un rôle dans la mise en place des frontières entre les organes latéraux et entre le méristème et les organes latéraux pendant le développement post-embryonnaire. Les tiges régénérées à partir de culture d'hypocotyles* du double mutant *cuc1 cuc2* forment des fleurs dont les organes sont fusionnés, indiquant que ces gènes sont également importants pour la mise en place des frontières des organes floraux (Souer *et al.*, 1996; Aida *et al.*, 1997; Ishida *et al.*, 2000). Cependant, ces mutations ne perturbent pas l'initiation des feuilles ou des méristèmes floraux, suggérant l'existence de gènes redondants.

Émergence des primordia. Une fois les cellules fondatrices recrutées et séparées du reste du méristème, l'organe commence sa croissance et émerge sur le flanc du méristème. Le gène *AINTEGUMENTA* (*ANT*) est un des principaux régulateurs de la croissance des organes latéraux connus chez *Arabidopsis*. Ce gène code pour un facteur de transcription de la famille APETALA2 et est un marqueur des primordia, où il s'exprime dès le recrutement des cellules fondatrices (Elliott *et al.*, 1996; Long et Barton, 1998). Les études sur le mutant *ant* semblent indiquer que le gène *ANT* pourrait réguler la prolifération cellulaire

en agissant sur la machinerie cellulaire. Ainsi, le mutant *ant* montre des organes floraux atrophiés, perturbant notamment la mise en place des téguments⋆. Le développement végétatif est moins perturbé mais les feuilles contiennent moins de cellules que celles de la plante sauvage (Elliott *et al.*, 1996; Klucher *et al.*, 1996; Krizek, 1999; Mizukami et Fischer, 2000). Inversement, une sur-expression du gène *ANT* augmente le nombre de cellules dans les organes et induit une expression ectopique⋆ de la cycline D3.

I.1.3 L'auxine

I.1.3.1 L'auxine dans la plante

L'auxine est la première hormone végétale à avoir été décrite et elle est semble impliquée dans tous les aspects de la croissance et du développement de la plante (Vernoux, 2002, chap. IV). Sous le nom "auxine" sont en fait regroupées un ensemble de molécules de petite taille et de structure relativement simple. L'auxine naturelle la plus abondante est l'acide indole-3-acétique (AIA⋆). Mais d'autres auxines ont été décrites : des auxines naturelles comme l'acide indole-3-butyrique 5 (AIB ; Epstein et Ludwig-Müller, 1993) ; ou des composés chimiques artificiels ayant une activité auxinique comme l'acide 2,4-dichlorophénoxyacétique (2,4-D) ou l'acide naphtalène acétique (ANA).

Les réponses physiologiques à l'auxine sont multiples. L'hormone est notamment impliquée dans la croissance différentielle des tissus lors de différents tropismes⋆ et notamment dans le gravitropisme racinaire et le phototropisme de la tige. Elle régule également l'élongation des tissus, la différenciation des tissus vasculaires, l'établissement de la dominance apicale, l'initiation des racines latérales et des primordia dans l'apex caulinaire⋆ (Davies, 1995). Au niveau cellulaire, l'auxine régule la division, l'allongement et la différenciation, avec des différences dans la sensibilité et la nature de la réponse à l'auxine selon les tissus. La diversité des rôles de l'auxine est illustrée par la variété des phénotypes des mutants affectés dans les étapes de la biosynthèse ou dans la signalisation de l'auxine (pour revue : Hobbie et Estelle, 1994; Bartel, 1997; Rogg et Bartel, 2001).

I.1.3.2 Synthèse et voies de signalisation de l'auxine

La quantité intra-cellulaire de l'auxine est régulée par une combinaison de quatre processus : la biosynthèse, la conjugaison, la dégradation et le transport. La biosynthèse de l'AIA⋆ semble assurée par plusieurs voies parallèles que nous ne détaillerons pas (pour revue : Bartel, 1997). L'auxine peut être conjuguée de manière réversible (Davies, 1995). Il semble que, sous sa forme conjuguée, elle ne soit pas active et qu'il puisse donc exister des réservoirs d'auxine inactive, les conjugués pouvant être hydrolysés si nécessaire. La dégradation peut se voir soit directement par une décarboxylation oxydative, soit par l'intermédiaire des conjugués (voir par exemple : Tuominen *et al.*, 1994).

La variété et la complexité des mécanismes de biosynthèse rend difficile l'étude des lieux de synthèse de l'auxine. L'hypothèse majoritaire place la production de l'auxine principalement dans l'apex caulinaire⋆ et les jeunes feuilles. L'auxine serait ensuite transportée dans le reste de la plante (Reinhardt *et al.*, 2003b).

Les voies de signalisation de l'auxine sont multiples et complexes. On sait depuis quelques années que l'auxine induit la transcription en facilitant la dégradation de membres

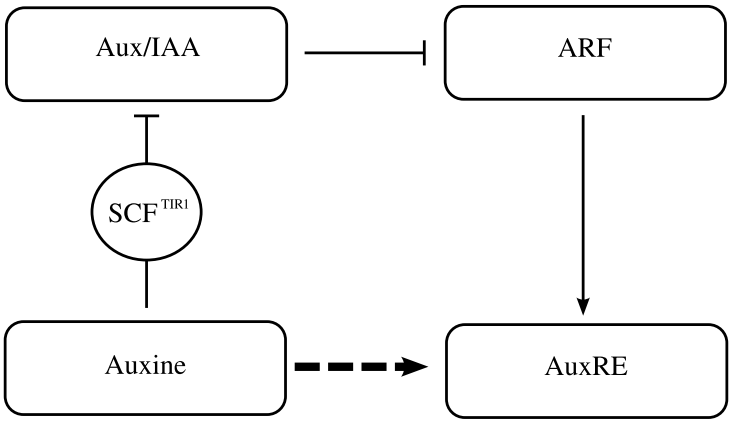


FIG. I.8: **Vue (simplifiée) d'une voie de signalisation de l'auxine.** Dans cette voie de signalisation, l'auxine se lie à la ligase⋆ SCF^{TIR1}, pour catalyser l'ubiquitination⋆ de répresseurs transcriptionnels de la famille Aux/IAA. Les facteurs de réponse à l'auxine (ARF) peuvent donc activer les éléments de réponse à l'auxine (AuxRE). Cette voie est donc une voie d'activation des AuxRE par l'auxine.

de la famille de répresseurs transcriptionnels Aux/IAA (Gray et Estelle, 2000). À leur tour, les protéines Aux/IAA peuvent se dimériser avec les facteurs de transcription de la famille des facteurs de réponse à l'auxine (ARF, auxin response factor), et donc réprimer leur activité (Tiwari *et al.*, 2001). La dégradation de Aux/IAA se fait par ubiquitination⋆, catalysée par un complexe protéique de type SCF ciblant l'ubiquitine⋆ (Gray *et al.*, 2001). Les complexes SCF ont une structure particulière en sous-unités et sont constitués en une protéine SKP1, une Culline, une protéine à F-box et RXB1 (Deshaies, 1999). La spécificité de la cible de la protéine SCF dépend de la protéine à F-box. Les protéines Aux/IAA d'*Arabidopsis*, sont ciblées par la protéine TIR1 et probablement par quelques autres homologues⋆ (Dharmasiri *et al.*, 2005). Il a été montré très récemment, que l'auxine agissait directement en se liant à TIR1, ce qui provoque la dégradation de Aux/IAA. On peut donc dire que TIR1 est un récepteur direct de l'auxine qui agit comme médiateur de la réponse transcriptionnelle de l'auxine (Figure I.8 : Dharmasiri *et al.*, 2005; Kepinski et Leyser, 2005).

Une fois l'inhibition levée, les gènes répondant à l'auxine peuvent être activés par des protéines toujours présentes dans la cellule. Les promoteurs de ces gènes incluent un motif spécifique appelé AuxRE ("Auxin Response Element"). Les techniques de génie génétique ont permis de construire, à partir de l'AuxRE du gène *GH3*, un promoteur synthétique bien plus réactif à l'auxine que tous les autres promoteurs connus. Ce promoteur est nommé *DR5* (Ulmasov *et al.*, 1997). Avec ce promoteur, il devient possible, en l'intégrant dans une construction avec une protéine fluorescente comme *DR5::GFP*, d'observer dynamiquement les zones de concentration d'auxine dans les plantes sauvages ou mutantes.

I.1.3.3 Transport de l'auxine

Une des spécificités de l'auxine par rapport aux autres hormones végétales est que sa diffusion dans la plante dépend largement d'un transport actif polarisé. Dans la partie aé-

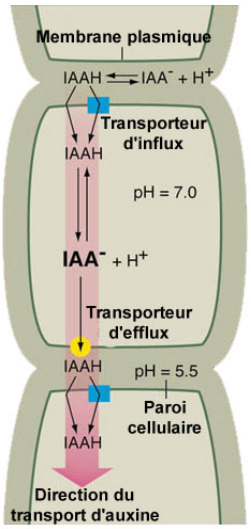


FIG. I.9: **Modèle chémiosmotique de transport d'auxine.** Le modèle chémiosmotique de transport suppose l'existence d'un transporteur d'efflux, qui permet à l'auxine de passer du compartiment intra-cellulaire à l'espace inter-cellulaire, et d'un transporteur d'influx, qui permet à l'auxine de passer de l'espace inter-cellulaire au compartiment intra-cellulaire.
Adapté de : Jones (1998)

rienne de la plante, l'auxine voyage principalement par les tissus vasculaires de l'apex vers les racines (Lomax *et al.*, 1995). Dans le système racinaire, l'auxine voyage essentiellement par les cellules de la stèle* puis, après avoir atteint l'apex racinaire, elle est redistribuée de la pointe de la racine vers le haut de la racine par les tissus épidermiques et corticaux.

Le modèle de transport de l'auxine tel qu'il est connu aujourd'hui est dit chémiosmotique et a été proposé par Rubery et Sheldrake (1974) et Raven (1975). Un modèle chémiosmotique propose l'existence d'un transporteur d'efflux et d'un transporteur d'influx (voir Figure I.9). En effet, la forme la plus courante de l'auxine, l'AIA* est un acide faible. À l'extérieur de la cellule, le pH étant faible, il est trouvé majoritairement sous sa forme protonée (AIAH). L'auxine protonée étant lipophile, elle peut diffuser à travers la membrane plasmique, mais un transporteur d'influx permet d'augmenter le flux entrant. Une fois dans la cellule, l'auxine à tendance à reprendre sa forme ionisé (AIA^-) et ne peut alors plus passer la membrane plasmique. Il lui faut un transporteur d'efflux pour sortir de la cellule. La polarisation du transport d'auxine est dû à une localisation polarisée du transporteur d'efflux.

Transporteurs d'influx. Chez *Arabidopsis*, le principal transporteur d'influx putatif caractérisé est AUX1 (Bennett *et al.*, 1996; Marchant *et al.*, 1999). Le gène *AUX1* code pour une protéine membranaire similaire à des transporteurs d'acides aminés. Ces transporteurs fonctionnent comme des symports* acide aminés-proton et le fait que l'auxine soit probablement cotransportée avec un proton a permis à Bennett *et al.* (1996) de proposer que AUX1 soit un transporteur d'influx de l'auxine.

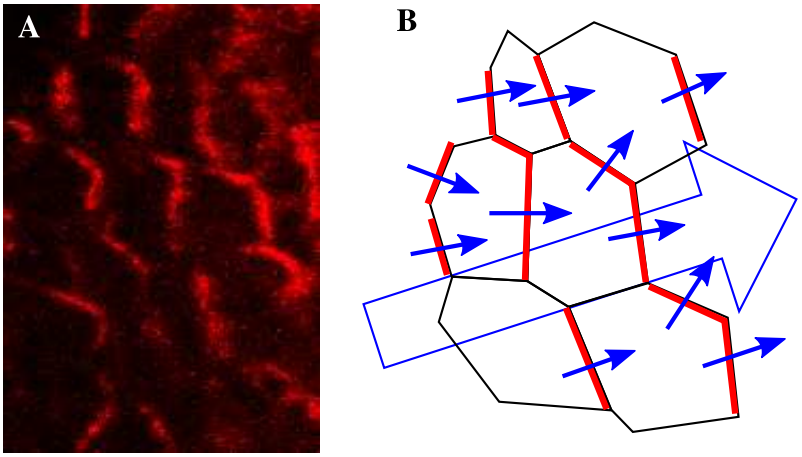


FIG. I.10: **Position polarisée de PIN1 dans le méristème.** Localement, la protéine PIN1 est polarisée de manière cohérente entre les cellules. Cette polarisation induit un flux d'auxine. La figure **A** montre un immunomarquage ciblant PIN1 issu d'un MAC. La figure **B** montre le modèle de flux qu'on peut en extraire. La large flèche bleue indique la direction globale du flux alors que les petites flèches indiquent la direction locale.
Image : Morin, H. et Traas, J., non publiée

L'implication de AUX1 dans le transport d'influx de l'auxine a été confirmé en analysant l'impact de l'ajout exogène de différentes auxines sur les racines du mutant *aux1*. En effet, les différentes auxines n'utilisent pas les différents transporteurs avec la même efficacité (Delbarre *et al.*, 1996). Par exemple, l'ANA est capable de diffuser beaucoup plus facilement que l'AIA* à travers la membrane plasmique sans transporteur d'influx. Inversement, le 2,4-D diffuse beaucoup plus difficilement et l'importance du transporteur d'influx est très grande. Or, l'ajout d'ANA permet de complémenter le phénotype du mutant *aux1*, indiquant clairement que AUX1 facilite l'influx d'auxine dans les cellules (Marchant *et al.*, 1999).

Transporteurs d'efflux. Le transport d'efflux de l'auxine est facilité (et probablement même assuré) par les protéines de la famille PIN. Ces protéines sont notamment sensibles aux inhibiteurs de transports que sont l'acide N-naphtylphtalamique (NPA) et l'acide triiodobenzoïque (TIBA). Ainsi, l'ajout exogène d'un de ces inhibiteurs suffit à rendre inopérant le transport d'efflux d'auxine (Morris *et al.*, 1991). Les différents membres de la famille *PIN* sont exprimés à des endroits différents de la plante. Par exemple, *PIN1* est exprimé dans la partie aérienne de la plante alors que *PIN2-4* sont exprimés dans le système racinaire.

Contrairement aux protéines AUX1, il semble les protéines PIN ne soient pas réparties uniformément dans la cellule, mais qu'elles définissent une polarisation de la cellule, orientant le flux d'auxine à travers celle-ci. Cette polarisation est le plus souvent localement cohérente entre les cellules, assurant un transport polarisé de l'auxine au niveau du tissus (figure I.10 ; Reinhardt *et al.*, 2000).

Rôle de la protéine kinase PINOID (PID). Les mutations perte de fonction *pid* produisent des phénotypes similaires à ceux des mutants *pin* (Bennett *et al.*, 1995). Principalement, les inflorescences du mutant *pid* sont très semblables à celles du mutant *pin1*, indiquant un rôle essentiel de ce gène dans la production des organes. Des études récentes ont montré l'implication de la protéine kinase⋆ PID dans le contrôle de la polarisation des protéines PIN (Friml *et al.*, 2004). Il semblerait que cette protéine fonctionne comme un interrupteur : en-dessous d'un niveau critique, PID induit une polarisation basipète de PIN alors qu'au-dessus elle induit une polarisation acropète.

I.1.3.4 L'auxine dans le MAC

Comme le montre les mutants *pin1* et *pid*, le transport de l'auxine est primordial pour la création des organes latéraux mais ne semble pas influer la maintenance du méristème lui-même. Nous allons maintenant décrire ce que nous savons sur le lien entre l'auxine et l'organogénèse.

Le traitement de l'apex avec de l'auxine exogène perturbe l'initiation des organes et modifie la phyllotaxie⋆ (Snow et Snow, 1937; Wardlaw, 1955a,b). Par exemple, Reinhardt *et al.* (2000) ont montré que l'application localisée d'AIA⋆ sur l'apex de la tomate à la position de l'initium I-1 provoque une augmentation de la taille du primordium. Aussi, l'application localisée d'AIA à la position de l'initium I-2 provoque la formation d'un méristème ectopique⋆ et l'inversion de la phyllotaxie.

Une expérience déterminante indiquant le rôle de l'auxine dans la formation des organes est celle réalisée par Reinhardt *et al.* (2000). Sur un mutant *pin1*, l'application localisée d'AIA⋆ non-loin de l'apex provoque l'initiation d'un primordium à proximité de l'application. Cette expérience a permis de proposer deux hypothèses : la première est qu'une concentration élevée d'auxine induit l'apparition d'un primordium, la seconde est que la protéine PIN1 est nécessaire pour concentrer suffisamment d'auxine pour produire un organe. Une autre étude utilisant le NPA pour mimer le phénotype de *pin1* a montré que plus la quantité d'auxine appliquée est importante, plus la probabilité d'initiation est grande, quel que soit le type d'auxine (Stieger *et al.*, 2002), ce qui semble confirmer un rôle d'interrupteur à primordium pour l'auxine dans le MAC. Par contre, l'initiation se fait systématiquement dans un anneau entourant la ZC du MAC, quel que soit le point d'application de l'auxine.

L'étape suivante a été de regarder précisément le transport d'auxine dans le méristème car l'action de l'auxine est très dépendante de sa concentration, elle-même dépendante de son transport.

L'immunomarquage⋆ de la protéine AUX1 montre qu'elle n'est présente que dans la *tunica* et principalement dans la L1, suggérant que l'auxine est majoritairement présente dans la couche L1 du méristème (Reinhardt *et al.*, 2003b). Cette impression est renforcée par l'immunomarquage de PIN1 qui n'est présent que dans la *tunica*, dans les primordia et le système pro-vasculaire. Par la suite, l'étude des flux d'auxine s'est donc limité à la couche L1 du méristème, ce qui a facilité largement les recherches.

Une étude détaillée des immunomarquages de PIN1 a permis de construire un modèle qualitatif de l'action de l'auxine dans le méristème. Dans ce modèle, l'auxine provient des parties basses de la plante et arrive jusqu'aux primordia où elle est évacuée. Un nouveau

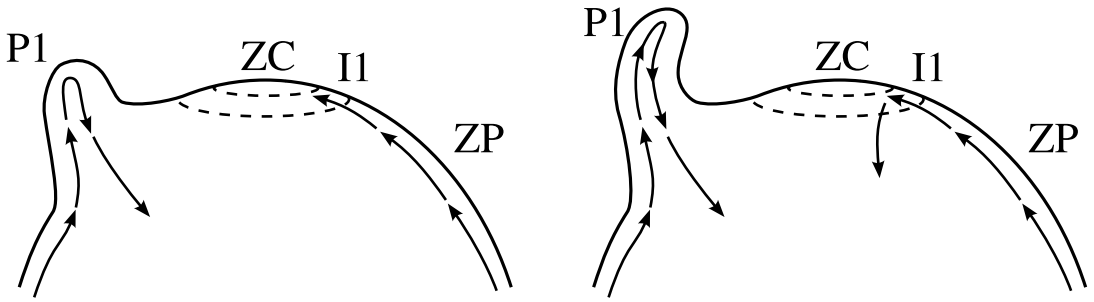


FIG. I.11: **Modèle qualitatif des flux d'auxines dans le méristème.** Dans ce modèle qualitatif, l'auxine est dirigée vers les primordia existant, d'où elle est évacuée vers le système pro-vasculaire. Cette évacuation empêche la formation d'un primordium à proximité d'un autre. Un primordium pourra se former en I1 si les autres sont suffisamment loin. Une fois formé, il va, à son tour, attirer et évacuer l'auxine, empêchant la formation d'un autre primordium dans son voisinage.

primordium ne peut alors se créer que loin des primordia existant (Figure I.11; Reinhardt *et al.*, 2003b).

I.2 Modélisation de la phyllotaxie

Dès le début du XIXème siècle, Schimper (1830) pose les bases de la phyllotaxie⋆ comme l'étude du positionnement des organes latéraux (et plus particulièrement des feuilles) pour son étude du méristème apical caulinaire⋆, alors appelé *punctum vegetationis*. Depuis, de nombreux travaux ont été menés pour modéliser la phyllotaxie. Ces travaux se classent en trois catégories :

1. les travaux purement descriptifs dont l'objectif est de caractériser mathématiquement l'arrangement géométrique des organes et d'étudier les liens entre les divers paramètres mesurables ;
2. les travaux sur les modèles dynamiques reposant sur l'hypothèse qu'un primordium inhibe la formation d'un organe dans son voisinage ;
3. les autres travaux sur les modèles dynamiques.

I.2.1 Modèles descriptifs

Les spirales phyllotaxiques. Dès les travaux de Schimper (1830); Braun (1831, 1835), l'étude de la phyllotaxie⋆ s'est portée sur la caractérisation des organisations spiralées et verticillées visibles sur beaucoup de plantes adultes. Il a notamment été considéré deux descriptions possibles pour des spirales. Dans les deux cas, les spirales sont dessinées en reliant par un trait continu les organes dans un ordre bien défini.

Si les organes sont reliés par ordre d'apparition, la courbe résultante forme une spirale (ou une hélice selon que l'on regarde la phyllotaxie⋆ sur un plan ou un cylindre) appelée *spirale génératrice*. L'angle formé par deux organes successifs sur cette spirale est le plus

souvent constant et est appelé *angle de divergence*⋆. Le plus souvent, l'angle de divergence⋆ est proche de l'angle d'or, $\phi = 2\pi(-1+\sqrt{5})/2$, où $(-1+\sqrt{5})/2$ est le nombre d'or (i.e. la racine de l'équation $x^2 + x - 1 = 0$).

Si chaque organe est relié à ses plus proches voisins ou, selon les descriptions, aux organes en contact avec lui, le tracé obtenu forme deux ensembles de spirales superposables à une rotation près (i.e. l'équivalent de parallèle pour des spirales) de sens de rotation opposés appelées les *parastiches*. Si i et j sont les nombres de spirales de chaque ensembles (avec $i < j$) alors le plus souvent, i et j sont deux nombres successifs de la suite de Fibonacci. Pour rappel, la suite de Fibonacci commence par les termes $u_0 = 1$, $u_1 = 1$ et a pour relation de récurrence $\forall n \in \mathbb{N}, u_{n+2} = u_{n+1} + u_n$. Les premiers termes sont donc : 1, 1, 2, 3, 5, 8, 13 ... Il est alors remarquable que le nombre d'or, noté φ, soit la limite à l'infini du rapport entre deux valeurs successives de la suite de Fibonacci :

$$\varphi = \lim_{n\to\infty} \frac{u_n}{u_{n+1}} = \frac{\sqrt{5}-1}{2} \tag{I.1}$$

Représentation ponctuelle des organes. Les frères Bravais (Bravais et Bravais, 1837a,b, 1839) proposent de représenter la position des feuilles comme un ensemble de points sur un cylindre et de le dérouler sur le papier, formant un treillis cylindrique (voir figure I.12). Cette représentation leur permet notamment de faire le lien entre la fraction continue⋆ de l'angle de divergence⋆ et les parastiches. Si on note $[a_0; a_1, a_2, \dots]$ la fraction continue de φ (le nombre d'or), les convergents principaux de la fraction continue sont les rationnels $[a_0; a_1, a_2, \dots, a_n]$ pour tout $n \geq 1$. Si on note $p_0, p_1, \dots, p_n$ les primordia, il est alors possible de retrouver la valeur de ϕ en comme le rapport des indices des primordia qui sont en contact avec p_0 au fur et à mesure que le nombre de parastiches devient infini (Jean, 1983, section 1.4.1).

Représentation volumique des organes. Airy (1873, 1874) a introduit l'utilisation de représentations volumiques des organes. Il propose une expérience simple permettant de produire les différents motifs phyllotaxiques : en disposant des billes sur un cylindre élastique adhérent étiré en deux rangées diamétralement opposées et décalées, le cylindre ayant été vrillé (voir figure I.13). En fonction des diamètres relatifs des billes et du cylindre, l'organisation billes quand le cylindre est relâché reproduits différents motifs phyllotaxiques. Aussi, en disposant plusieurs billes à la même altitude sur le cylindre équiréparties sur la circonférence d'une section, il est aussi possible de reproduire les modes verticillés. Ce modèle permet de donner une interprétation de la phyllotaxie comme un arrangement de billes limité par leur encombrement spatial. La construction fait donc intervenir un phénomène physique et non une construction purement mathématique.

La phyllotaxie comme empilement de disques. Inspiré par le travaille de Airy, van Iterson (1907) a étudié en détail les motifs obtenus en empilant des disques souples de taille variable sur la surface d'un cylindre (voir figure I.14 (A)). Il a ainsi montré que pour les phyllotaxies spiralées, un seul paramètre permettait de contrôler la phyllotaxie : le rapport b du diamètre des disques représentant les organes et du diamètre du cylindre représentant la tige. Ainsi, pour chaque valeur de b, seuls un certain nombre d'angles de

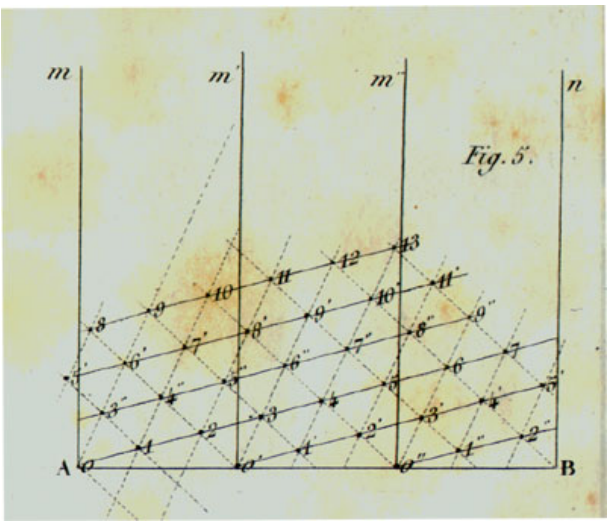


FIG. I.12: **Treillis cylindrique de la phyllotaxie⋆.**Bravais et Bravais (1837a) proposent de représenter la position des feuilles comme un treillis cylindrique pour étudier leur agencement. Un treillis cylindrique correspond au déroulé du cylindre représentant la tige répliqué trois fois. Ainsi, dans cette représentation, les droites (O, m), (O', m'), (O'', m'') et (B, n) correspondent en fait à une même et unique droite sur la tige répliquée pour mieux présenter la relation d'adjacence entre les points. Chaque point est ainsi répliqué 3 fois, ce qui permet de voir ici, par exemple, l'adjacence entre 3 et 5 qui sont pourtant de par et d'autre de l'origine choisie. En plus de permettre de mieux voir la relation d'adjacence en tout point de la tige, cette représentation représente les hélices phyllotaxiques comme des droites, sur lesquelles il est plus facile de travailler (parallélisme, comptage, ...).

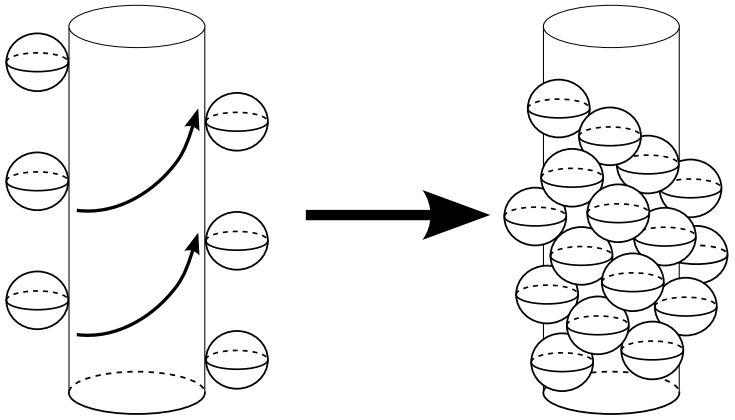


FIG. I.13: **Expérience de Airy (1873).** Airy propose de disposer des sphères sur un cylindre élastique préalablement étiré selon la direction indiquée par les flèches (i.e. extension et torsion combinée) en deux rangées parallèles diamétralement opposées et décalées (dessin de gauche). Après avoir relâché le cylindre, les billes se retrouvent en contact les unes avec les autres et sont placées selon un motif phyllotaxique.

divergence* α sont possibles. Au fur et à mesure que le paramètre b diminue (donc que la taille des primordia devient petite par rapport à la taille de la zone centrale), le nombre possible d'angles de divergence* augmente. $\alpha(b)$ a une forme d'arbre binaire dans lequel chaque branchement correspond à un changement de phyllotaxie (voir figure I.14B).

De nombreux autres résultats. Par la suite, Irving Adler, Roger V. Jean, Ralph O. Erickson et beaucoup d'autres ont précisés les propriétés mathématiques liées à la phyllotaxie. On peut notamment citer les liens entre l'angle de divergence* et les parastiches mais aussi la formalisation de la notion de parastiches visibles, l'étude de l'arbre formé par les angles de divergences possibles en fonction du rapport des diamètres, etc. (pour revue : Jean, 1983; Adler *et al.*, 1997).

Enfin, très récemment, Cummings et Strickland (1998) ont pu reconstruire les différents modes phyllotaxiques classiques (spiralés, verticillés, décussés) à partir de la minimisation d'une fonction d'énergie traduisant de façon abstraite l'interaction de deux morphogènes*. Le point intéressant est que le résultat n'est pas numérique mais analytique et qu'il leur permet de proposer une construction géométrique de la phyllotaxie.

La théories des hélices foliaires multiples. Au milieu du XXème siècle, Lucien Plantefol propose une théorie phyllotaxique radicalement différente de toutes les précédentes. Dans son ouvrage *Fondements d'une théorie phyllotaxique nouvelle - La théorie des hélices foliaires multiples* (Plantefol, 1948), il dénonce les théories existantes comme incomplètes ou infondées. Au lieu de s'appuyer sur la position des feuilles uniquement, il propose de prendre en compte trois éléments dans l'étude de la phyllotaxie :

1. **les insertions foliaires**, ou cicatrices foliaires, avec leur position et, chose nouvelle, leur forme exacte ;
2. **les segments foliaires**, qui correspondent à la partie de la tige qu'on peut associer à une feuille donnée (voir figure I.15) ;
3. **les faisceaux foliaires**, fibres internes à la tige qui passent de feuille en feuille ;

et de décrire la phyllotaxie comme les rapports existant entre ces différentes entités.

En étudiant essentiellement les insertions foliaires, il dessine des hélices allant de feuille en feuille, reposant sur le principe que, "prolongées sur la tige par leurs segment foliaire, les feuilles apparaissent *contiguës* sur une même hélice" (voir figure I.15 ; Plantefol, 1948, p. 145). Il propose, pour justifier l'existence de ces hélices, que chacune d'elle soit créée par un *centre générateur* différent. Chaque centre générateur crée un organe hélicoïdal continue le long duquel se répartissent les feuilles. Ils seraient coordonnés par un *centre organisateur*. Dans ce modèle, il est intéressant de noter que les hélices sont premières et que les feuilles sont des organes disposés sur les hélices, alors que dans les autres modèles, les feuilles sont premières et les hélices ne sont qu'une description possible de l'organisation des feuilles.

I.2.2 Modèles dynamiques historiques

Les modèles présentés dans cette section et dans la suivante ajoutent aux précédents la notion de création des primordia. Il ne s'agit plus d'étudier le résultat de l'organogénèse

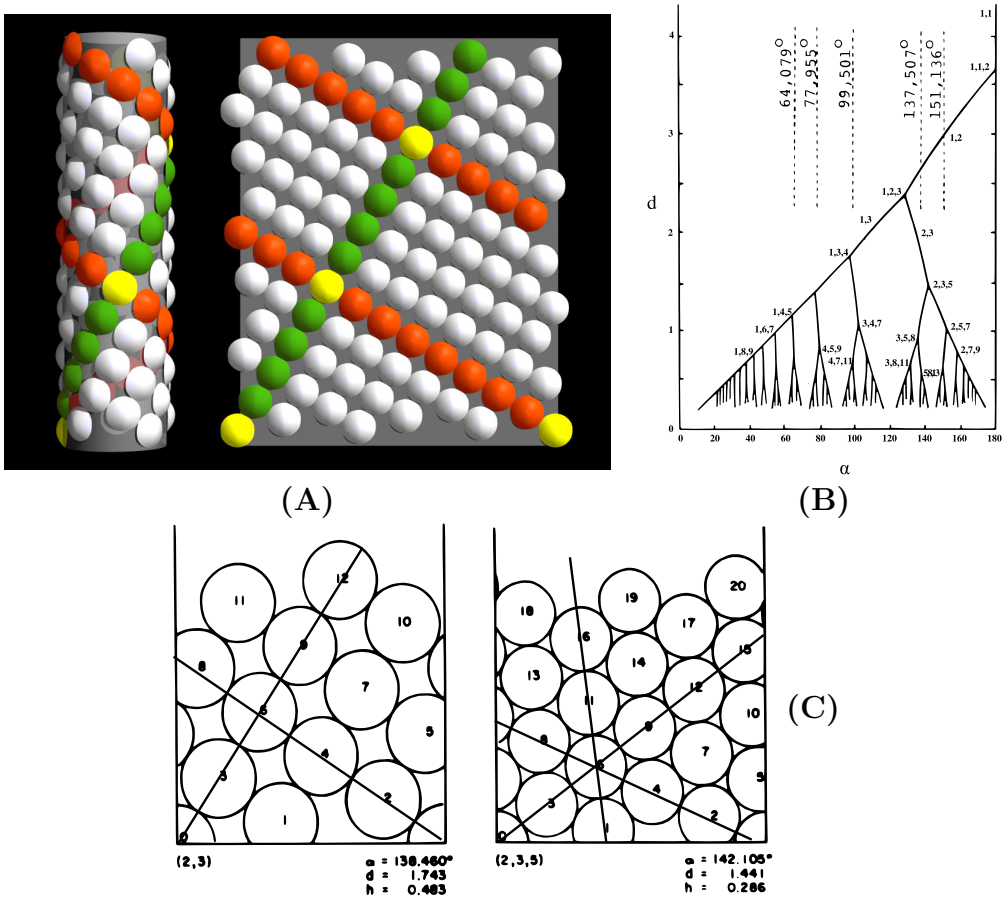


FIG. I.14: **Empilement de disques souples à la surface d'un cylindres (van Iterson, 1907).** **(A)** présente un empilement de disques souples à la surface d'un cylindre, le cylindre représentant la tige de la plante et chaque disque représentant un organe. En orange une hélice gauche et en vert une hélice droite s'intersectent aux organes indiqués par des disques jaunes. **(B)** Avec en ordonnée le diamètre des disques et en abscisse l'angle de divergence* (en degré), ce graphe montre les différents modes phyllotaxiques possibles. Les nombres de parastiches sont indiqués sur les principales branches et les principaux points triples. **(C)** Le construction géométrique permet de comprendre pourquoi il y a des bifurcations et comment la phyllotaxie évoluent à ces endroits du graphe. Le point clef est le passage d'une phyllotaxie à deux ensembles d'hélices parallèles (cf. droites tracées pour l'organe 6 à gauche) à une phyllotaxie à trois ensembles d'hélices parallèles (cf. droites tracées pour l'organe 6 à droite). Quand le diamètre des disques diminue, dans une phyllotaxie (i, j), le primordium $n+i+j$ se rapproche du primordium n. Le diamètre devient suffisamment petit pour que ces deux primordia soient juste en contact. La phyllotaxie devient alors triple en $(i, j, i+j)$. Si le diamètre diminue encore, le primordium $n+i+j$ reste en contact avec le primordium n soit en écartant le primordium $n+i$, ce qui donne une phyllotaxie $(j, i+j)$ soit en écartant le primordium $n+j$, ce qui donne un phyllotaxie $(i, i+j)$.
(A) : Prusinkiewicz et Lindenmayer (1990)
(B) : Jean (1983), version adaptée de la schématisation de Veen et Lindenmayer (1977) des tracés de van Iterson (1907).
(C) : Veen et Lindenmayer (1977)

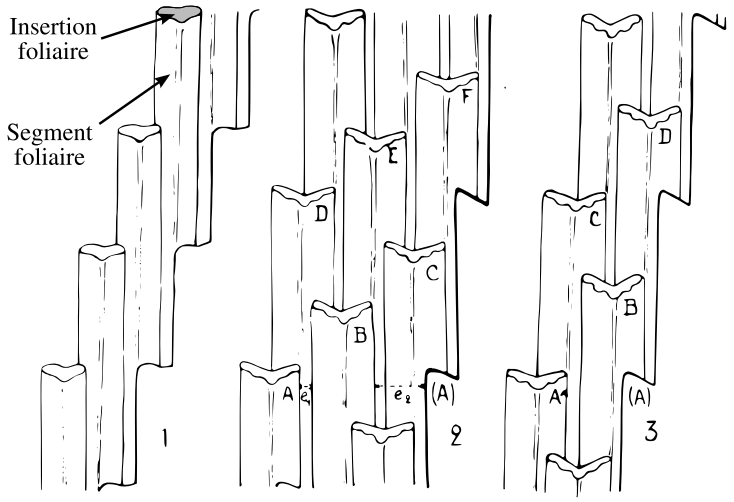


FIG. I.15: **Hélices foliaires sur *Lilium tigrinum*.**(Plantefol, 1948) Ce schéma montre les insertions foliaires et les segments foliaires qui leurs sont attachés. Chaque tige a été "dépliée" de façon à ce que le bord gauche épouse la forme du bord droit. Les hélices sont visible en passant d'un segment foliaire à l'autre uniquement s'ils sont en contact direct. 1. Tige à une seule hélice foliaire. 2. Tige à trois hélices foliaires. 3. Tige à deux hélices foliaires.

mais le processus qui peut mener à ce résultat. Dans cette section, nous nous intéresserons aux modèles découlant d'observations morphologiques uniquement, macroscopiques ou microscopiques. Les modèles utilisant les données de la biologie cellulaire, moléculaire et de la génétique seront étudiés dans la section suivante.

Hypothèse des champs d'inhibition. Dès 1868, Hofmeister propose un modèle dynamique extrêmement robuste permettant de produire les phyllotaxies spiralées (Hofmeister, 1868, source : Douady et Couder (1996a)). Dans ce modèle, il suggère que chaque primordium émet un champs d'inhibition additif qui gène la formation d'un nouveau primordium et que les primordia s'éloignent du centre avec une vitesse radiale $V(r)$ (voir figure I.16). Une horloge interne à la plante provoque la création d'un primordium a intervalle de temps régulier. Celui-ci se forme sur un cercle de rayon R_0 à la position où le champ d'inhibition des primordia déjà présents est minimum. Avec ces règles très simples, il est possible de recréer toutes les phyllotaxies spiralées (Douady et Couder, 1996a).

Douady et Couder (1992) proposent une expérience physique permettant de reproduire une phyllotaxie avec un système similaire. Dans cette expérience, une assiette en Téflon posée horizontalement, remplie d'huile de silicone est plongée dans un champ magnétique vertical. Ce champ magnétique a un faible gradient radial : il est minimum au centre et maximum à la périphérie. L'apparition d'un primordium est simulée par l'ajout d'une goutte de ferrofluide au centre de l'assiette. La goutte se polarise alors pour former un dipôle vertical, qui est attiré vers les zones où le champ magnétique est le plus fort, assurant une vitesse radiale du "primordium" limitée par la viscosité du silicone. Comme les gouttes forment des dipôles parallèles, elles se repoussent les unes les autres avec un force proportionnelle à d^{-4}, d étant la distance entre deux gouttes. Enfin, un dispositif

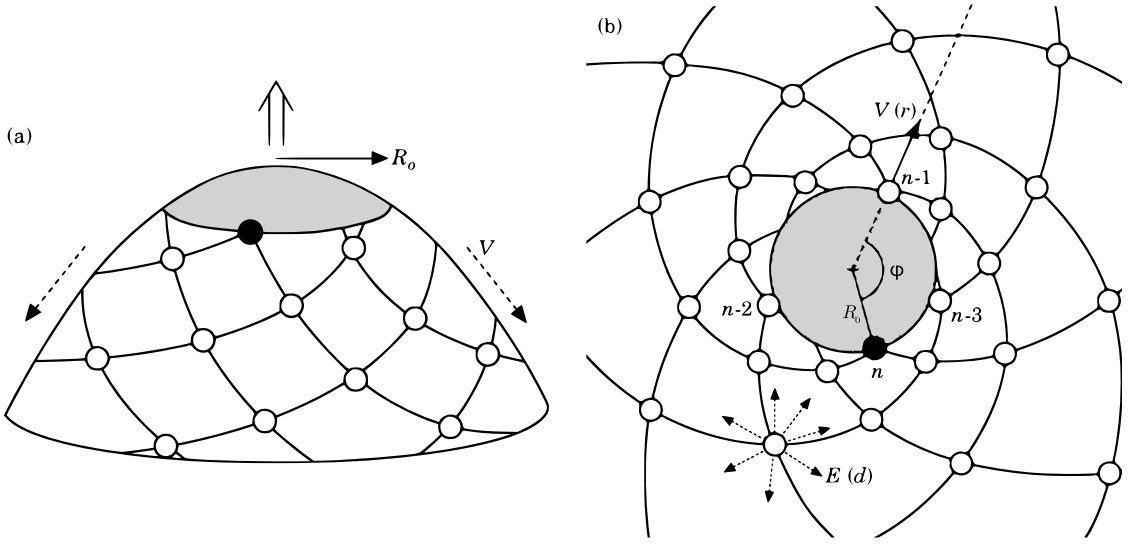


FIG. I.16: **Définition des paramètres de la croissance apicale.** Chaque primordium s'éloigne du centre du méristème à une vitesse $V(r)$ du centre et émet un champ d'inhibition $E(d)$. Quand un nouveau primordium apparaît, c'est toujours au minimum du champ d'inhibition résultant des primordia existant sur un cercle de rayon R_0. L'angle formé entre deux nouveau primordia est noté ϕ est appelé angle de divergence*.
Images : Douady et Couder (1996a)

permet de déposer des gouttes calibrées à un rythme régulier réglable pour permettre l'ajout de primordia à intervalle de temps constant (voir figure I.17).

Ce système ne reproduit pas fidèlement le comportement des primordia dans un méristème. Notamment, les gouttes ajoutées interagissent avec les gouttes déjà présentes, modifiant leurs positions de façon non négligeable. Malgré tout, de nombreux motifs phyllotaxiques spiralés peuvent être obtenus par ce système. Un résultat intéressant est que le motif ne dépend que d'un seul paramètre : $G = V_0T/R_c$ où R_c est le rayon de stabilisation angulaire des gouttes (i.e. leur position angulaire ne change plus significativement au-delà), V_0 est la vitesse des gouttes à la distance R_c du centre et T est l'intervalle de temps entre deux lâchés de gouttes.

Pour étudier plus en détail les conséquences de ce jeu d'hypothèses, Douady et Couder (1996a) ont recouru à la simulation. Ils ont alors pu tracer, comme l'avait fait van Iterson (1907) pour des empilement de disques, les phyllotaxies possibles en fonction de ce paramètre G (voir figure I.18). Il est intéressant de noter que certains angles de divergences qui sont considérés comme possibles par les empilements de disques ne le sont pas par cette méthode dynamique (voir dans la figure I.18 les zones où le trait plein n'est pas recouvert par les triangles noirs, notamment aux alentours de $\varphi = 120$).

Cet ensemble d'hypothèses pose comme acquis la régularité d'apparition de nouveaux primordia. L'inconvénient est que l'obtention de phyllotaxies verticillées est alors interdite, à moins d'imposer le nombre de primordia simultanés. Comme il est courant de voir des changements de nombre de verticilles au cours de la vie de la plante (voir figure I.19), la nécessité de rajouter un paramètre variable pour rendre compte de ce seul changement

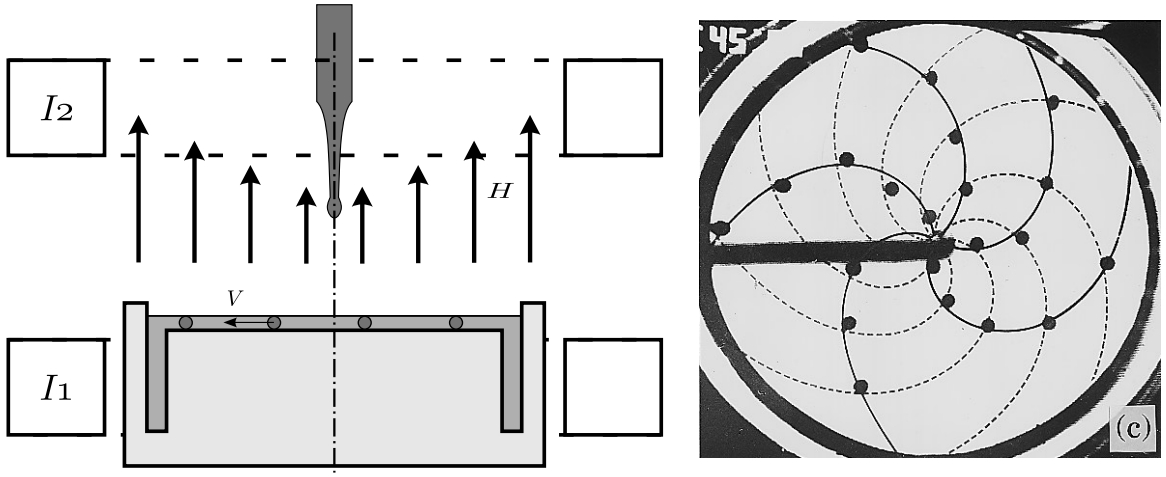


FIG. I.17: **Production d'une phyllotaxie par des particules magnétisées.** Des gouttes de liquide ferro-magnétique sont lâchées au centre d'une assiette remplie d'huile de silicone et plongée dans un champ magnétique vertical (image de gauche). Le dispositif permet de lâcher des gouttes calibrées à un rythme régulier réglable. Ce dispositif permet de générer divers motifs phyllotaxiques spiralés (image de droite).
Images : Douady et Couder (1996a)

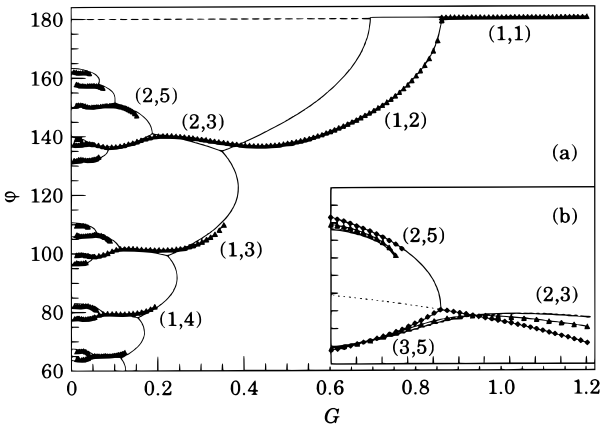


FIG. I.18: **Phyllotaxies générées en introduisant les primordia régulièrement.** En abscisse le paramètre G (représentant le rapport de taille entre les primordia et le méristème, ce qui est équivalent à la définition de G présentée dans le texte) qui permet de résumer les réglages, et en ordonnée l'angle de divergence. Ce graphique montre les angles de divergences possibles. En trait plein sont représentées les phyllotaxies possibles obtenues par empilement de disques et les triangles représentent les phyllotaxies obtenues par simulation dynamique avec une loi d'interaction $E(d) = 1/d^3$. En (b), détail d'une bifurcation obtenue en changeant la loi d'interaction entre les primordia $(-)$ $E(d) = 1/d$, $(\triangle)$ $E(d) = 1/d^4$, $(\diamond)$ $E(d) = exp(-d/l)$ avec $l = 0.01$.
D'après : Douady et Couder (1996a).

FIG. I.19: **Changement du nombre de verticilles de 3 à 4 sur une tige d'*Abelia.*** Image : Douady et Couder (1996c)

n'est pas très satisfaisant, d'autant plus que les changements de phyllotaxie à nombre de verticilles constant ne nécessitent pas ce paramètre.

Hypothèse du premier espace disponible. Une modification des hypothèses, notamment proposée par Snow et Snow (1937) permet de pallier ces problèmes. Au lieu de supposer un intervalle constant d'apparition des primordia, ils supposent qu'un primordium se positionne dès qu'il y a suffisamment de place. Ceci se traduit dans le modèle précédent par un seuil σ tel que si la répulsion des primordia existant devient inférieure à σ en un point du cercle d'apparition des primordia, alors un primordium apparaît en ce point (voir figure I.20 pour un modèle plus simple).

Il est intéressant de remarquer que la régularité d'apparition des primordia découle naturellement de ces hypothèses si l'état stable est atteint. De plus, les modes verticillés et les transitions entre eux apparaissent tout aussi naturellement. Par contre, une étude exhaustive des modes possibles pour différents paramètres devient extrêmement complexe.

Douady et Couder (1996b) ont réalisé une étude systématique des états stables et des transitions entre états stables sous cette hypothèse. Ils ont ainsi montré que toutes les formations connues (spiralées, verticillées, opposées décussées) pouvaient être obtenues sans autre hypothèse. Mais l'étude la plus probante confirmant ce modèle est probablement celle des transitions entre les modes phyllotaxiques stables. Il a en effet été montré que les transitions prédites par ce jeu d'hypothèses reproduisent fidèlement les positions observées dans la nature, notamment sur les feuilles tournesol, dont les positions exactes sur la tige étaient réputées imprévisibles car elles ne suivent pas un motif phyllotaxique classique (i.e. spiralé, verticillé...) mais semblent disposées aléatoirement (Douady et Couder, 1996c; Couder, 1998).

Modèles de diffusion. Les modèles de diffusion sont très proches des modèles faisant l'hypothèse du premier espace disponible. Les primordia sont souvent représentés par des points à la surface d'un cylindre (Young, 1978; Chapman et Perry, 1987). Ils diffusent

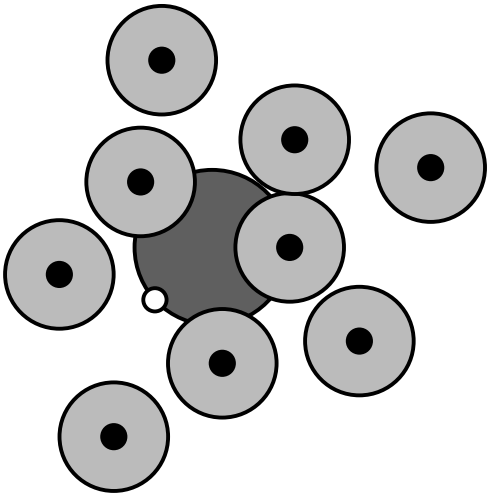


FIG. I.20: **Zones d'inhibition des primordia.** Le disque central représente le méristème avec les primordia qui se forment à sa périphérie. Les disques noirs représentent les primordia, entourés de leur zone d'inhibition. Un nouveau primordium peut se former à la périphérie du méristème dès que sa zone d'inhibition n'intersecte aucune autre. Ainsi, le prochain primordium apparaîtra à la position du disque blanc.
Adapté de : Steeves et Sussex (1989)

un inhibiteur qui se dégrade au cours du temps, empêchant la formation d'un nouveau primordium dans le voisinage des primordia existants. Le centre aussi diffuse un inhibiteur, parfois le même (Young, 1978), parfois un inhibiteur indépendant (Chapman et Perry, 1987).

Chapman et Perry (1987) proposent de remplacer l'inhibiteur produit par chaque primordium par un activateur consommé par ceux-ci. On suppose alors qu'un activateur est normalement présent uniformément dans la tige de la plante, et qu'il est consommé par les primordia. Un primordium se forme alors en un point ayant une concentration suffisante d'activateur. Mathématiquement, supposer la consommation d'un activateur est équivalent à supposer la production d'un inhibiteur. Toutefois, cette hypothèse est plus satisfaisante d'un point de vue biologique, car il existe un bon candidat pour l'activateur dans les protéines végétales : l'auxine, qui est connues pour promouvoir l'apparition des primordia (Reinhardt *et al.*, 2000).

I.2.3 Modèles dynamiques originaux

Modèle biomécanique de Green. Paul B. Green, un biomécanicien de l'université de Standford, propose un modèle basé sur la déformation mécanique de la surface du méristème (Green *et al.*, 1996; Green, 1992, 1999).

Son modèle repose sur les ondulations produites par la croissance des parties internes d'un anneau. En effet, si on fixe les bords d'un anneau mais que le milieu croît, des oscillations de longueur d'onde déterminée par les propriétés mécaniques du matériau constituant l'anneau se forment (voir figure I.22A-F). Pour générer des motifs phyllotaxiques, il propose de calculer sur un anneau les oscillations dues à la croissance. Puis, pour simuler

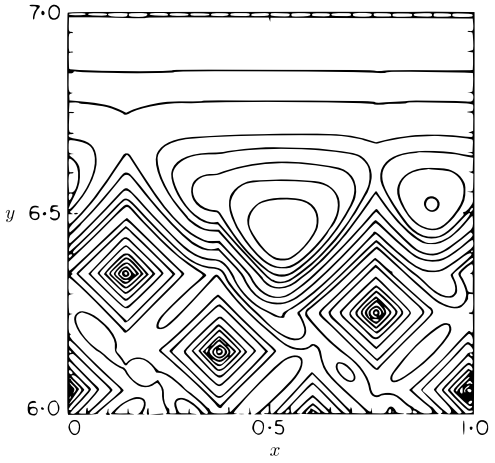


FIG. I.21: **Courbes de niveau de la concentration d'inhibiteur.** Cette image montre le développé d'un résultat obtenu par Young (1978) d'une phyllotaxie (2,3) sur un cylindre. Le méristème est à la position $y = 9.606$ (au-dessus du graphique) et donne naissance aux contours rectilignes en haut de la figure. Les maxima sont situés au centre du méristème et au centre des successions de carrés concentriques dénotant la présence d'un primordium. Le prochain primordium apparaîtra sur le minimum qui peut être vu au centre de la figure, et la position du primordium suivant peut déjà être observée sur la droite.
Image : Young (1978)

la croissance de l'apex, l'anneau est d'abord figé, puis agrandi par homothétie. Enfin, un anneau interne est ajouté. Les oscillations de l'anneau externe servent de conditions aux limites pour le bord extérieur de l'anneau interne (le bord interne étant figé) pour le calcul des nouvelles oscillations (voir figure I.22G). Par ce biais, il obtient tout un panel de phyllotaxies, spiralées ou verticillées, en fonction des propriétés mécaniques du matériaux simulé et en fonction de la variation de diamètre des anneaux.

Patrick D. Shipman et Alan C. Newell ont repris, précisé et complété les travaux de Green. Ils ont ainsi largement étudié la production des différents modes phyllotaxiques, les transitions entre modes phyllotaxiques et la relation entre la phyllotaxie et la forme des apex (Shipman et Newell, 2004, 2005).

Modèle de réaction-diffusion de Meinhardt. Hans Meinhardt propose un modèle largement inspiré de son étude des coquillages et des motifs qu'on y trouve (Meinhardt, 2003a,b). Ainsi, il propose un modèle de plante dans lequel la croissance est produite par accrétion cellulaire au sommet de la tige.

Dans ce modèle, un processus de réaction-diffusion évolue sur un anneau de cellules. À intervalle de temps régulier, l'anneau de cellules se dédouble et le rang inférieur devient inactif, figeant la concentration des morphogènes*. La phyllotaxie est obtenue en regardant la position géométrique des pics de concentration d'un des morphogènes, considéré comme le marqueur de position des primordia (voir figure I.23 A,B).

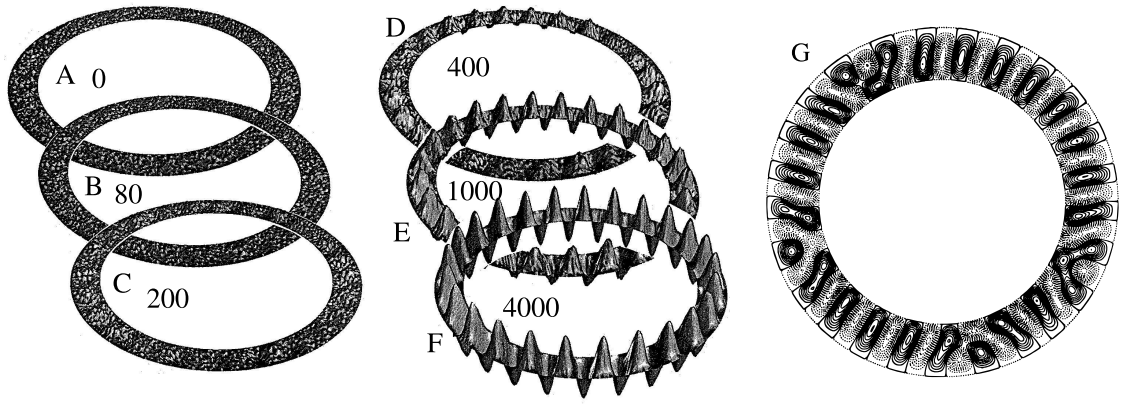


FIG. I.22: **Déformations mécaniques dans un anneau. (A-F)** Un anneau de diamètre interne 0.8 et de diamètre externe 1.0 est créé avec des ondulations aléatoires à sa surface. Ses propriétés physiques donnent une longueur d'onde intrinsèque pour les ondulations de 0.2. Les deux bords sont fixés. Les différentes images montrent la position de minimum d'énergie au fur et à mesure que la partie interne (i.e. entre les bords de l'anneau) du tissus croît (le nombre dans les anneaux est proportionnel à la quantité de matière ajoutée). Il est intéressant de voire que l'apparition des ondulations est irrégulière (C,D) mais que par la suite, les amplitudes et les distances se régularisent (E,F).
(G) Pour produire la phyllotaxie, la rangée extérieure d'ondulation est fixée alors que la rangée interne est laissée libre. La configuration montrée correspond au minimum d'énergie. La différence de diamètre entre l'intérieur est l'extérieur étant importante, on observe un changement du nombre d'ondulations avec des figures caractéristiques en Y.
Images : Adaptées de Green *et al.* (1996)

Le modèle consiste alors en un système de réaction-diffusion capable de générer des pics de morphogène* à intervalle de temps régulier et à des positions vraisemblables pour la phyllotaxie. Meinhardt y parvient en utilisant un système à trois morphogènes :

1. un activateur auto-catalytique[3] a, qui joue le rôle de marqueur pour les primordia
2. un substrat b qui est consommé par l'activateur
3. un inhibiteur c ayant une durée de vie assez longue mais qui diffuse peu, même si plus que l'activateur

Les équations différentielles régulant les concentrations de morphogène*s sont :

$$\frac{\partial a}{\partial t} = \frac{sb\left(a^2+b_a\right)}{\left(c_c+s_c c\right)\left(1+s_a a^2\right)} - r_a a + D_a \frac{\partial^2 a}{\partial x^2} \tag{I.2}$$

$$\frac{\partial b}{\partial t} = b_b - \frac{sb\left(a^2+b_a\right)}{\left(c_c+s_c c\right)\left(1+s_a a^2\right)} - r_b b + D_b \frac{\partial^2 b}{\partial x^2} \tag{I.3}$$

$$\frac{\partial c}{\partial t} = r_c a - r_c c + D_c \frac{\partial^2 c}{\partial x^2} \tag{I.4}$$

Les deux derniers termes de chaque équation correspondent respectivement à la dégradation et à la diffusion du morphogène. Le premier terme correspond, lui, à la production du morphogène et aux interactions avec les autres morphogènes. Ainsi l'équation régulant la variation de l'activateur inclue un terme d'auto-catalyse. Celle-ci est conditionnée à l'existence de substrat (b) et est limitée par une borne supérieur ($1 + s_a a^2$ au dénominateur) et par la production d'inhibiteur ($c_c + s_c c$ au dénominateur). Le substrat quant à lui est produit régulièrement (premier membre de l'équation) mais est consommé d'une quantité égale à l'auto-catalyse de l'activateur (deuxième membre). Enfin, l'inhibiteur est simplement produit proportionnellement à la quantité d'activateur présent (premier membre de l'équation).

Dans ces équations, r_i, D_i et b_i sont les coefficients de dégradation, diffusion et production du morphogène* i. s est le coefficient d'auto-catalyse de l'activateur. c_c a pour rôle d'empêcher une surproduction de l'activateur quand il n'y a plus d'inhibiteur. s_c correspond à l'efficacité de l'inhibiteur. Enfin, s_a définit la limite supérieure de l'auto-catalyse de l'activateur.

Le couple activateur/substrat permet d'obtenir des figures oscillantes comme des vagues qui se déplacent. L'ajout d'un inhibiteur à longue durée de vie ($b_c << b_a$) a pour rôle de diminuer les concentrations d'activateur peu de temps après son maximum. Si cet inhibiteur longue durée a une portée plus grande que l'activateur ($D_c > D_a$), alors le pic d'activateur disparaît complètement avant de réapparaître à un autre endroit, là où le substrat est en quantité suffisante pour activer la réaction. L'activateur semble alors sauter d'un lieu à l'autre. Si l'inhibiteur reste présent suffisamment longtemps, il empêchera l'activation au même lieu même après plusieurs apparitions de primordium. Cette inhibition locale dans l'espace mais à longue portée dans le temps permet notamment d'obtenir des primordia disposés selon un motif phyllotaxique spiralé (voir figure I.23 B-F).

[3] i.e. dont l'activation est régulée positivement par sa propre concentration

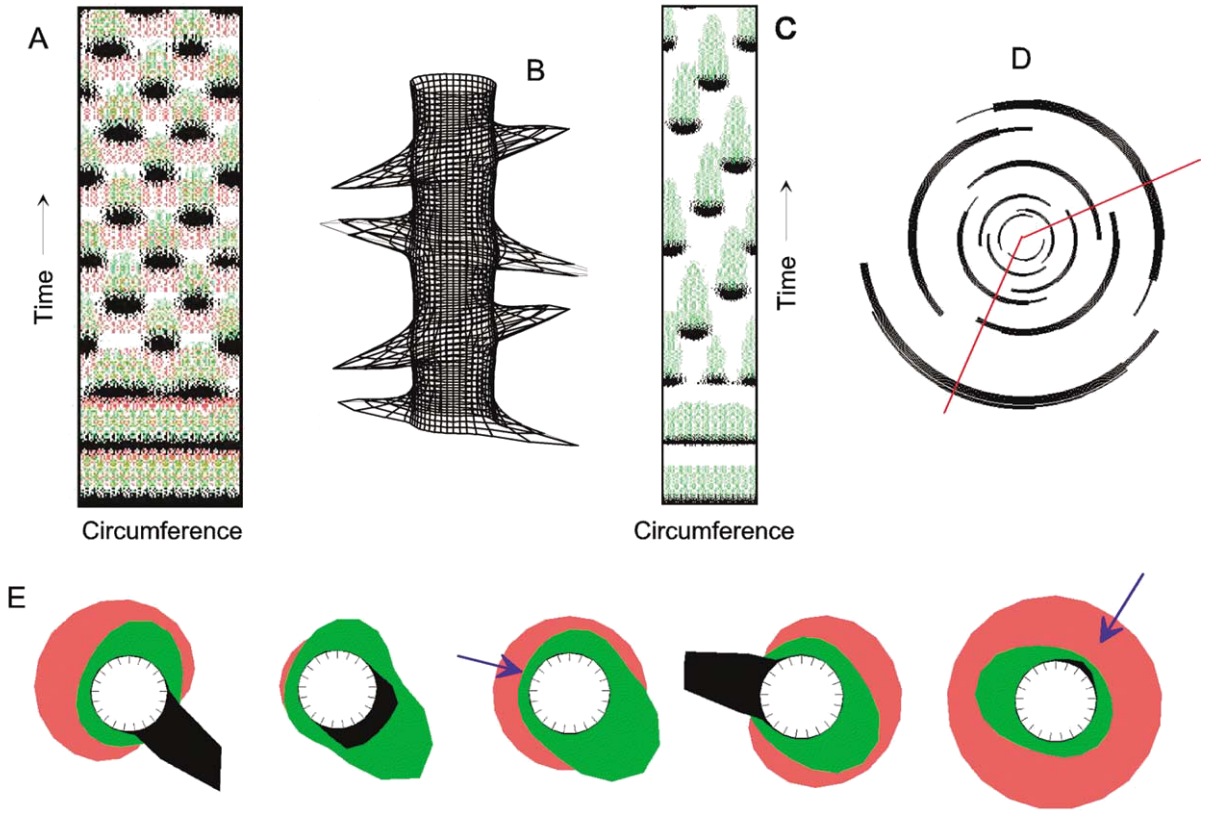


FIG. I.23: **Production d'une phyllotaxie par réaction-diffusion.** Toutes les simulations sont le résultat du modèle de réaction-diffusion à trois morphogènes* décrits dans le texte.
(A,B) si l'inhibiteur n'a pas une durée de vie assez longue, la figure obtenue correspond à une phyllotaxie verticillée opposée/décussée.
(C,D) mais en augmentant sa durée de vie, il est possible d'obtenir une phyllotaxie spiralée.
(E) Distributions des trois morphogènes à divers instants. Le noir donne la concentration de l'activateur, le vert de l'inhibiteur et le rose du substrat. La flèche indique le lieu d'apparition du prochain primordium. Un primordium peut se former si le niveau d'inhibiteur est suffisamment bas et s'il y a suffisamment de substrat.
Figures : Adaptées de Meinhardt (2003b)

Chapitre II

Reconstruction 4D de la surface d'un méristème

II.1 Résumé de l'article

La microscopie *in vivo* permet de générer des images sur le comportement dynamique d'objets tri-dimensionnels (3D). L'extraction de l'information utile dans ces images étant une tâche complexe, il est nécessaire de développer des outils mathématiques et informatiques adaptés pour aider à leur interprétation. Idéalement, une chaîne informatique complète pour l'étude de la dynamique d'objet tri-dimensionnels devrait inclure : l'acquisition (1), le préprocessing (2) et la segmentation (3) des images, suivis par une reconstruction dans le temps et l'espace de l'objet étudié (4) et enfin des analyses quantitatives (5). Ici, nous avons développé un tel protocole pour étudier la dynamique du méristème apical caulinaire d'*Arabidopsis*. Ce protocole utilise des piles de sections optiques obtenues par microscopie confocale. Il inclus notamment des algorithmes spécialement développés pour automatiser l'identification des lignées cellulaires et l'analyse quantitative du comportement de la surface du méristème.

Les auteurs de l'article sont, dans l'ordre : Pierre Barbier de Reuille, Isabelle Bohn-Courseau, Christophe Godin et Jan Traas.

Cet article a été publié dans le numéro 44, volume 6 de *The Plant Journal* dans les pages 1045 à 1053 sous le titre "A protocol to analyse cellular dynamics during plant development".

II.2 Introduction

The spectacular advances in imaging techniques have greatly contributed to our understanding of many cellular and molecular processes. With the advent of methods that allow the production of massive amounts of data, however, the imaging field is now facing new requirements. These do not only concern the quality and resolution of the pictures, but also the quantity of the information to be analysed. This is especially true for techniques involving the observation of living tissues. *In vivo* confocal microscopy imaging, allows the analysis of cell shape in space and time in intact tissues (Van den Berg *et al.*,

1995; Reddy *et al.*, 2004; Grandjean *et al.*, 2004) In addition, this technique can be combined with GFP technology, thus allowing the observation of specific cellular compartments or gene activity. These methods have opened up a wide range of new possible applications, but they also pose the problem of image and data analysis. Indeed, a major challenge is now to develop the tools to extract pertinent information on the dynamic spatial behaviour of the components that constitute a tissue. To study the dynamics of complex 3D structures various types of algorithms must be used. In principle, an complete procedure should include (1) the acquisition and (2) initial preprocessing of the images, (3) image segmentation, (4) three dimensional (3D) and four dimensional (4D) reconstruction and (5) the quantitative analysis. The acquisition of the images involves the generation of sets of serial optical sections. This is accompanied by an initial preprocessing (step 2) required to retain as much information as possible. This can vary from a simple setting of contrast to a non-linear filtering to remove unnecessary background noise. The third step, image segmentation, is required to extract and to identify the areas to be analysed. More specifically, this implies a classification of the pixels within an image, with their coordinates, as being part of the background or part of the object. For a structure divided into cells, this would typically imply the identification of the pixels corresponding cell membranes or nuclei for example. Subsequently, the 3D structure of the object is represented in such a way that specific quantitative spatial and temporal characteristics can be extracted, such as changes in cell size, cell shape, neighbourhoods etc. Most of the existing commercial softwares offer the possibility to reconstruct 3D objects from serial sections or to extract quantitative information (e.g. on cell size or shape) from the images. However, these tools are usually not able to generate the full protocol able to cover all 5 steps mentioned earlier.

In this paper, we show how image processing and data analysis can be combined to build up a software chain for 3D and 4D reconstruction. This includes the design of special algorithms to automate the complex processing of time series. To illustrate the approach, we have chosen the shoot apical meristem (SAM) in *Arabidopsis* thaliana. Shoot apical meristems (SAMs) are small groups of dividing stem cells that initiate all the aerial parts of the plant (Traas et Doonan, 2001). Over the last years an important amount of information has been accumulated on this structure, and *in vivo* techniques to study SAMs in the confocal microscope are now available. It therefore seemed particularly appropriate to develop a full 4D image analysis protocol for this system. The approach, however, can be extended to other structures as well.

II.3 Results

To visualise the meristems, we used a protocol previously described by Grandjean *et al.* (2004). This method allows the visualisation of living meristems at the cellular level in the confocal microscope. The cells either express GFP, and can thus be directly visualised, or are stained first using the fluorescent membrane stain FM 4-64, which permits the visualisation of cell contours and facilitates the reconstruction of cell arrangements in the meristem.

II.3.1 Step 1 and 2: acquisition of images and initial preprocessing in the microscope

Stacks of serial sections were taken at different time points. As the shoot meristem is a flattened structure, transverse sections allowed more easily a complete overview of the meristem than longitudinal sections. The standard software of the microscope was sufficient to guarantee an optimal acquisition of the fluorescent signals (i.e. to obtain images with the proper contrast, representing a maximal amount of relevant signals avoiding saturation or underexposure).

According to the confocal microscope software, sections should be made ideally every $0.36\mu m$. This distance is linked to the resolution of the confocal microscope along the focal axis. The resolution is determined using the optical laws governing confocal microscopy and mainly the diffraction pattern (see Webb, 1996). However, this implied that at least 100 sections had to be made of every meristem. Since this required an extensive exposure of the tissue to the laser beam, we also tested the quality of the reconstruction when less sections were taken (see also material and methods). This showed that up to $1\mu m$ between the sections, the quality was still excellent (i.e. 91% of the surface could be reconstructed). Above this value the errors rapidly increased. For example, at $1.8\mu m$ only 63% of the meristem surface could be digitised. In particular at the edge of the meristem the individual cells were no longer clearly distinguishable. Nevertheless, even at a distance of about $2\mu m$ the results were acceptable, in particular where the cells at the top of the meristematic dome were concerned. In general, a distance between the sections of $1\mu m$ appeared to be a good compromise.

After acquisition the images were further preprocessed to reduce background noise. This consisted first in the application of a small Gaussian filter of radius between 1 and 5 pixels to smooth images, in particular to reduce the impact of isolated bright or dark dots. In addition, a threshold was applied, removing all the pixels with a brightness less than an empirically defined constant (we used 10 for images with 256 grey levels). This was followed by a contrast optimisation to ensure that grey levels ranged from 0 to 255.

II.3.2 Step 3: Image segmentation: retrieval of geometry and topology

From the stacks of sections, the geometry and the topology of the cells of the surface of the meristem were extracted. In this context, geometry refers to the cell shapes, whereas topology characterises the neighbourhood relationship (adjacency) between the cells.

The reconstruction of the surface was done in two phases: (i) a reconstruction of the image of the meristem surface from the stack of images, (ii) definition of the topology and geometry of the cells in 2D.

II.3.2.1 Reconstruction of the image

The aim was to produce an image of the meristem seen from the top. Since the meristem is a dome with small bumps there is no structure that over shades other parts of the surface. As a consequence the top view permits visualisation of the whole meristem surface. To compute such a view, the parts corresponding to the meristem proper has to be separated

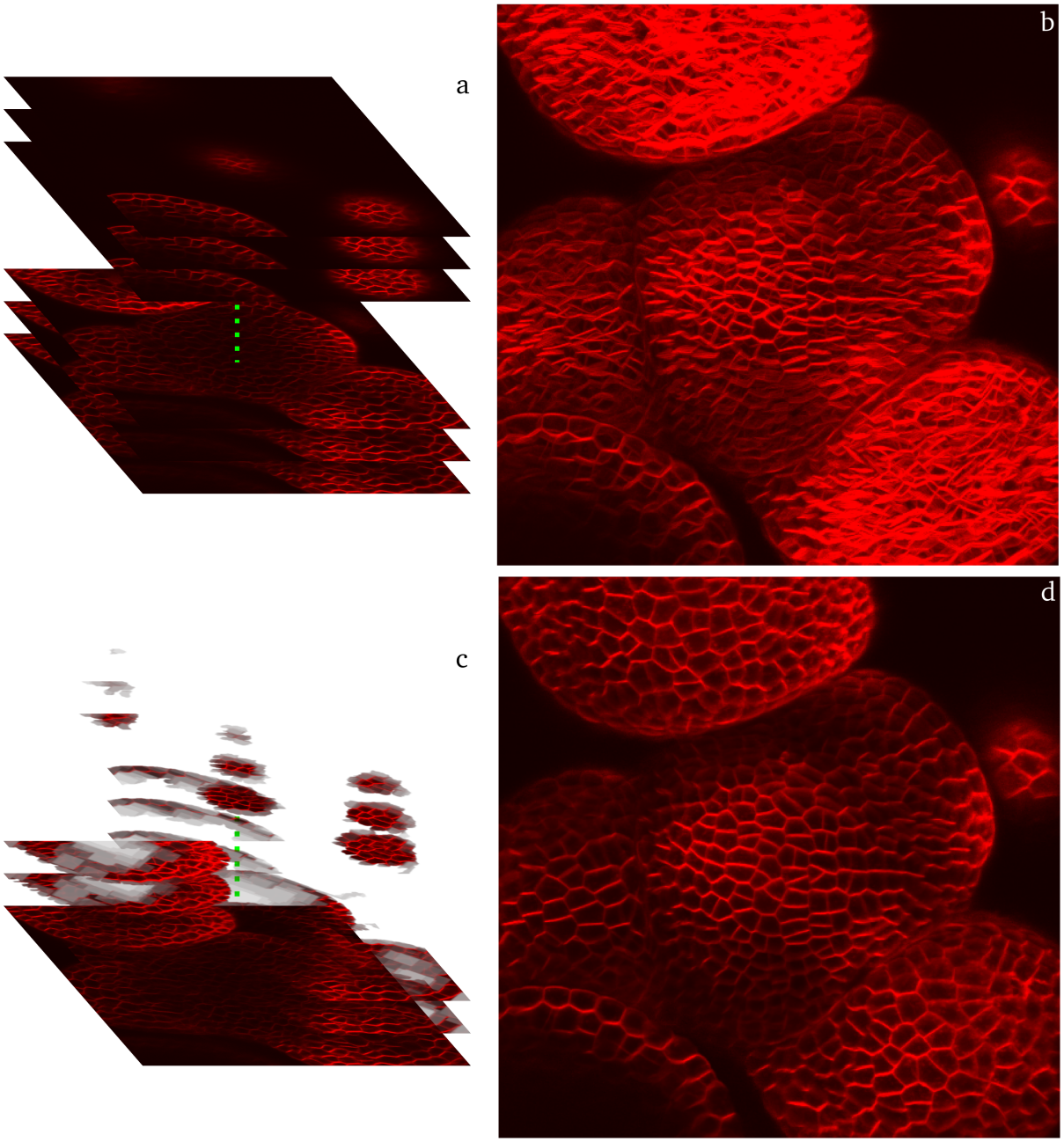


FIG. II.1: **Meristem surface reconstruction: general principle.** The confocal microscope produces a stack of serial sections of the observed meristem (a). The microscope's software allows a simple projection (b). In this projection, each pixel at position (x, y) is given the highest value of all pixels at position (x, y) in the image stack. The method described in this paper allows the reconstruction of the top view of the meristem surface (d). It first makes the parts outside the meristem transparent, as shown in c. Then, it constructs a top view of the stack of partly transparent images superposed on a black background. As a result, the surface layer of the meristem can be clearly distinguished.

from the background in each image of the stack (Figure II.1). This was achieved by making the area surrounding the meristem transparent, using a so-called transparency mask (see Supplementary material), while the image of the meristem proper was kept opaque. The image of the meristem surface was finally obtained by looking at the stack of images combined with their transparency mask from the top (Figure II.1). Note that, in Figure II.1c, the masked images used intermediate levels of transparency. We found that this allowed a better rendering of the surface image than with binary transparency without loosing useful information (see Supplementary material).

To compute the transparency masks, we needed to separate the areas outside from the areas inside the meristem. The problem was that they both contained black pixels. Figure II.2a shows a typical image from a stack of sections. In this image, cell walls are very bright while the cell interior and plant exterior are both dark. Fortunately, dark areas within the plants were smaller than the black areas outside the plant. We exploited this structural difference to fill in the smaller dark areas using the topological closure algorithm (see Serra et Vincent, 1992; Heijmans *et al.*, 1992; Vincent, 1992) and Figure 1 in Supplementary material). This technique fills in gaps having a width smaller than a given threshold. This allowed us to fill in the black pixels corresponding to the interior of the cells. Yet, in our context, this approach had a major drawback: when a separated primordium was too close to the meristem in an image (i.e. closer than the distance threshold, as shown by the white arrow in image II.2b), the gap between them was filled in as well. To overcome this limitation, we combined the topological closure with a segmentation technique called watersheds (Figure II.2).

A grey scale image (Figure II.2a) can be interpreted as a height map (the higher the intensity of the colour, the higher the point is, see Figure 2 in Supplementary material). In such a 'mountain' map, the interior of every cell is a valley (or watershed) between mountain chains. The dark area outside the meristem forms one or a few very large valleys. A comprehensive definition and an efficient algorithm for watersheds as seen in topology can be found in Vincent et Soille (1991).

After identification of the watersheds in the images (Figure II.2c), they were combined with the topological closure. For this purpose all watersheds covering black pixels in the topological closure image were identified. These were considered as being located outside the meristem and subsequently eliminated (i.e. set to transparent).

As explained above, the images of the stack were superposed with their transparency masks to reconstruct the top view of the meristem.

II.3.2.2 Definition of 2D geometry and topology

The goal of the reconstruction was to represent the geometry and the topology of the cells in the L1 layer of the meristem. This was achieved with assistance of computer tools specifically developed for this purpose.

Definition of vertices at wall intersection. Cell geometry was most easily defined by identifying manually the points where the walls intersect. Usually, these so-called vertices corresponded to points where three cells were in contact. Sometimes, however, four cells were in contact or two vertices were apparently so close, that they could not be distinguished. In this case we considered that four cells were in contact.

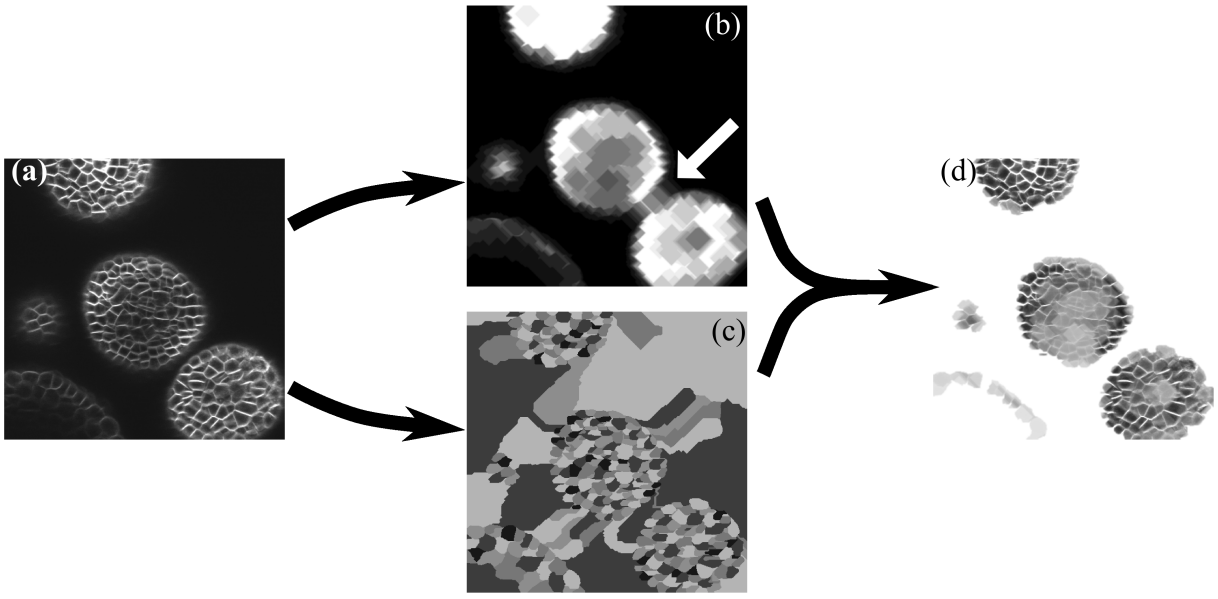


FIG. II.2: **Computation of the transparency mask.** From the original grey scale image (a) the topological closure is first computed. (b). This technique allows the identification of most of the areas outside the meristem (in black). The major problem with this technique is that it can lead to the apparent connection between the meristem and one of its primordia (white arrow). To correct this, the watersheds are computed. These are shown in (c), where the watersheds are represented as areas with different colours. The area outside the meristem can include several watersheds because of the background noise. The two methods are then combined. Hereby, first all watersheds covering at least one black pixel from the topological closure image are identified. These are considered to be located outside the meristem and entirely set to transparent. The transparency of the other parts is computed according to the topological closure (d).

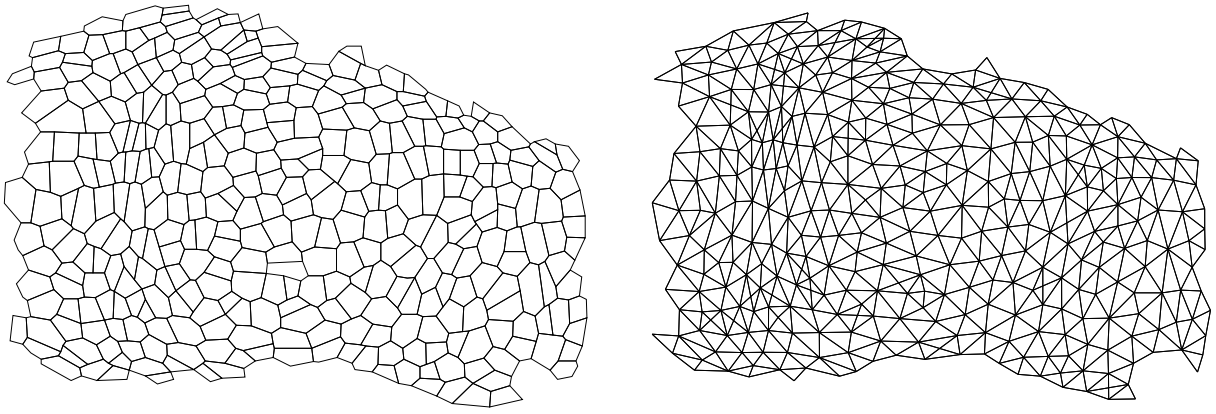


FIG. II.3: **Geometrical and topological representations of the meristem.** The left image represents the reconstructed 2D meristem with the cell walls (i.e. the geometry). The right image represents the centres of the same cells linked when they share a wall (i.e. the topology).

Definition of cell geometry. We chose to represent the geometry of cells by polygons. To define the polygons corresponding to the individual cells, the vertices had to be grouped and ordered. The vertices of individual cells were first grouped manually. To avoid the manual specification of order between vertices, a hypothesis regarding cell shape was made so that the geometry of the cells could be directly inferred from the positions of the vertices. We therefore made the assumption that cell polygons were star-shaped around their centre of gravity. A geometric object is star-shaped around one point p if, for every point p' of the object, the straight segment $[pp']$ is entirely contained within the object. So far, all observed cells could be considered as such.

Definition of cell topology. The groups of vertices corresponding to the cells were then used to compute cell topology. Two cells are neighbours if they have at least one vertex in common. The neighbourhood relationship was represented as a graph (Figure II.3b), where nodes corresponded to cells, while the links between nodes represented the adjacency between cells.

II.3.3 Step 4: Reconstruction in space (3D) and time (4D)

II.3.3.1 3D reconstruction

In step 3 the vertices were identified on a 2D projection (top view) of the dome. In step 4, these vertices were located in the 3D space by identifying them in the original stack of confocal images. Each vertex corresponded to a bright pixel (with coordinates (x, y)) in the 2D top view. Its z coordinate was determined by identifying the section of the stack intersecting the meristem surface with the coordinates (x, y). This section is the highest one having a pixel at position (x, y) with a brightness close to the one of the 2D projected surface image (Figure II.4). The brightness b of a point in the stack of images was considered close to the brightness b' of a point on the reconstructed projection

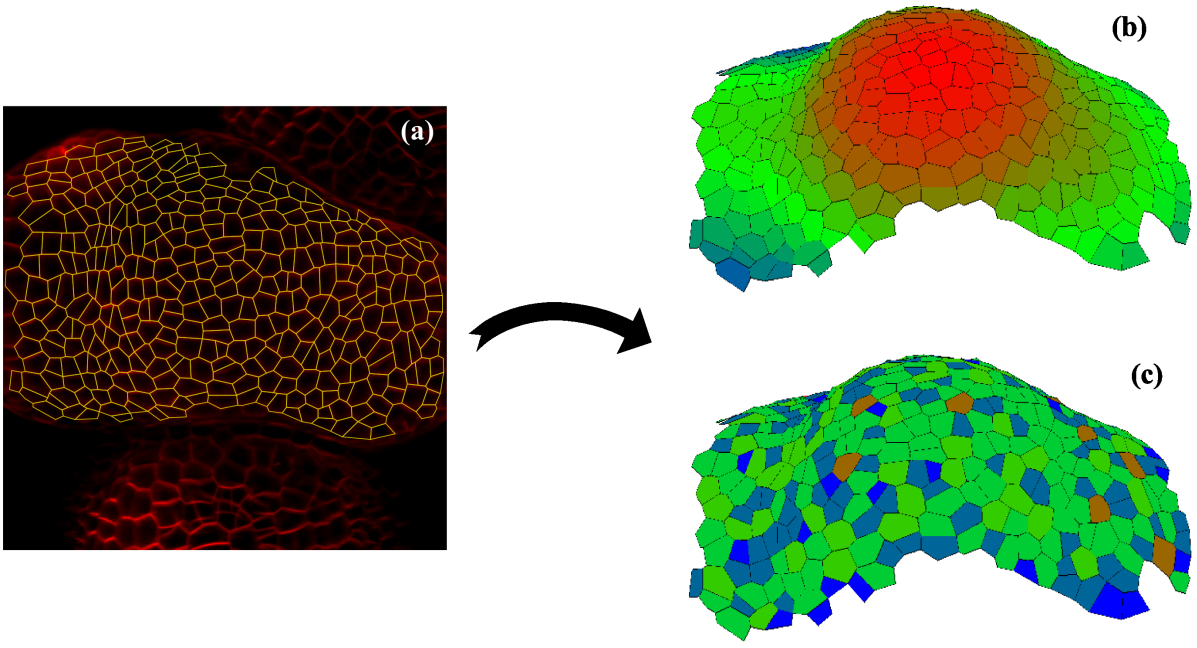


FIG. II.4: **Reconstruction of 3D surface of the meristem.** The 3D reconstruction of the surface was obtained using the original image stack, the reconstructed top view and the topological and geometrical 2D reconstruction(a, for details see text). Once the z coordinates identified, different types of information can be easily extracted, like the z coordinates of the cell centres (b) or the number of neighbours of each cell (c). In (b) blue stands for the lowest z, red for the highest and green for medium positions. In (c), cells are colour coded according to the number of neighbouring cells (dark blue corresponds to 4 neighbours, blue to 5 neighbours, green to 6 neighbours, light green to 7 neighbours and brown 8 neighbours).

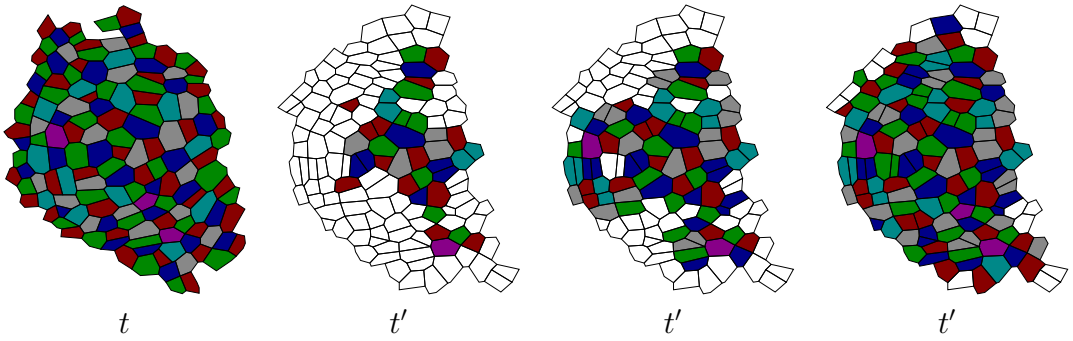


FIG. II.5: **Steps in cell associations.** The figure at the left represents the meristem at time t while the three other figures represent the same meristem at time t'. The algorithm identifies identical cells or mother cells and daughter cells at the two time points. The colour is kept constant for a given cell lineage.

if $b > \alpha b'$, where α is an experimental constant in $]0, 1]$ (in the current implementation $\alpha = 0.75$ gave excellent results).

II.3.3.2 Cell lineage identification

To analyse the dynamical behaviour of the tissue, meristems were observed at different time points and the confocal data were used to reconstruct their surface in 3D. As outlined above, these 3D reconstructions were defined as graphs representing both cell geometry and topology. Cells and daughter cells were initially associated manually in the successive reconstructions. However, since this task was very time consuming we developed an algorithm that semi-automatically followed the cell lineage in a meristem throughout time.

This program requires the manual association of one cell and one of its vertices in the first reconstruction with the corresponding cell and one of its vertices in the second reconstruction. It then goes from neighbouring cell to neighbouring cell, comparing cell shapes and vertex numbers. The combination of these two criteria allows the identification of most of the cells at successive time points and is able to recognises cell divisions. The principle of the algorithm is given in material and methods in computing details are in online material.

II.3.3.3 3D repositioning

Initially, every meristem was reconstructed in the reference system of the confocal microscope (i.e. in the reference system of the stack of images). As the meristems were taken out of the microscope and reinserted between two observations, we had to correct for artefactual reorientation. For this purpose, we readjusted the position of the meristem reconstruction for each new observation.

For this, we chose a reference system in which the tip of the meristem did not move and in which there is no global rotation of the meristem. The meristem centre was determined visually, based on morphological criteria, in particular with regard to the position of the primordia. For simplicity, the reference system was the one of the microscope at time t_0

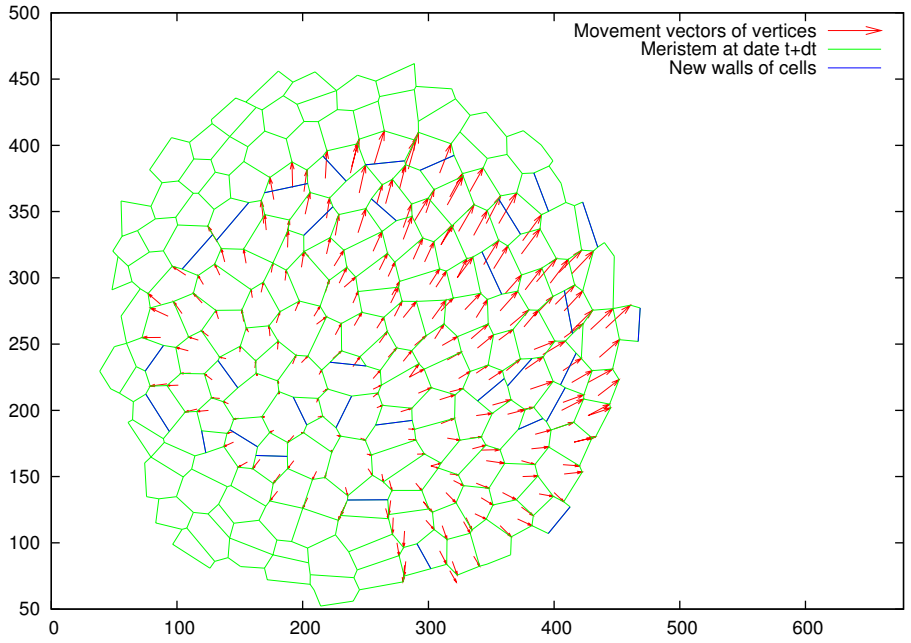


FIG. II.6: **Speed field computed after 3D repositioning.** Once the successive reconstructions of a meristem are in the same reference system, it becomes possible to compute the movement of the vertices in this reference system.

(i.e. the time of the first representation of the meristem). As the experimental conditions and the microscope settings were supposed to remain constant through the different observations of a given meristem, the transformation between the microscope and the meristem reference systems was at most the combination of a rotation and a translation. This transformation was computed using a min-square algorithm (see Supplementary material for details).

II.3.4 Step 5: Quantitative analysis

Since the reconstructions contain all the data regarding neihbourhood relationship, absolute position of the vertices and polygon representing the cells, a range of quantitative data can be easily extracted. Since a comprehensive analysis would be beyond the scope of this technical advance article, only a few examples are given in Figure II.4 and Figure II.6 where z position, the number of neighbouring cells, and speed fields in single meristems are given.

II.4 Discussion

To build up our software chain, we had to design and develop new algorithms. In addition, existing algorithms were used and assembled to be adapted to the specificity of our data (e.g. watersheds and filtering algorithms). As a consequence, a key issue in the system design was to integrate all these algorithms within a common toolkit with

flexible user interface. We chose to develop a software platform, based on a script language (Python) for which many extension libraries in image processing and data analysis exist. Dumais and Kwiatkowska used a similar approach to reconstruct and analyse the surface of the meristem from resin replicas observed in the electron microscope. However, their protocol for surface reconstruction was different from ours as it was based on stereo images of entire meristems and not on serial sections. Therefore, their protocol is not applicable to our data.

Haseloff and colleagues developed a software chain for the reconstruction of embryonic tissues and root meristems (see website `http://www.plantsci.cam.ac.uk/Haseloff/Home.html`) based on an existing image processing software. Their protocol allows for the reconstruction of geometry and topology of cells in intact tissues. However, their protocol requires a very high definition of the images, an optimal amount of sections and specific stains to differentiate different parts of the cells (Haseloff, 2003). Therefore fixed material and post-fixation treatments have to be used to improve the final image quality. The images obtained from the living meristems using our method clearly did not have the same resolution. This is not only because the staining method results in more background noise, but also because the number of sections through the living meristems has to be kept to a minimum. Therefore, we developed a different approach. It is true that our method is comparatively time consuming. This disadvantage, however, is compensated by an increased robustness.

An aspect that has not been considered in this study is the 3D rendering of internal parts of the meristem. Even if the 3D geometry of cells can be approximated by 3D polyhedrons, it is difficult to define the sides and vertices of the internal cells visually and manually. With the aim of developing an automatic (or semi-automatic) algorithm we made some preliminary studies on a completely automatic way to detect cells and extract their geometry, but this turned out to be a very difficult task. The method proposed by Haseloff and colleagues allows a semi-automatic reconstruction of the geometry and topology of the cells. As pointed out above, however, the reconstruction algorithm works on high resolution images of fixed material, which is, in principle, not applicable to our data. By any means, this type of semi-automatic approaches requires the manual identification of the cells, which is a time consuming task, in particular if time series of individual meristems are to be studied. We therefore conclude that at this stage a comprehensive analysis of cell shape dynamics in the internal parts of the meristem would require substantially more work.

II.4.1 Sources of error

Once the 4D reconstructions have been obtained, it can be difficult to recognise, *a posteriori*, artifacts that have occurred during the procedure and that can influence the final analysis. It is therefore important to identify such potential sources of errors at the moment they occur.

The first source of error is due to the confocal microscope itself. In first approximation, the obtained image of the meristem is not the meristem itself but the meristem convoluted by a function (called Point Spread Function). This function is governed by wave optics and is the cause of the resolution limits of the microscope (Webb, 1996). These limits,

in turn, cause a certain imprecision in the positioning of the vertices. The imprecision is greater along the z axis (i.e. along the focal axis) compared to the x and y axis (i.e. in the focal plane). The error in the focal plane can lead to an apparent merging of vertices that are very close to each other. This subsequently results in the generation of reconstructions where certain vertices are formed by four different cross walls. However, this only concerns a limited number of cells and has not been a major source of artifacts. In more general terms the errors generated by the resolution limits are well under the typical size of a cell, and do not pose a major problem in the final quantitative analysis. A potentially important source of error is a shape distortion of the objects due to optical aberrations or problems with the scanning device. Fortunately, these can easily be identified by using fluorescent beads with known size. A second problem along the z-axis is related to the thickness of the sections and the distance between them, which can lead to a miss-localisation of the vertices in the final reconstruction. In the case of one optical section every $2\mu m$ (approximately), this error can be up to $1\mu m$ at worst.

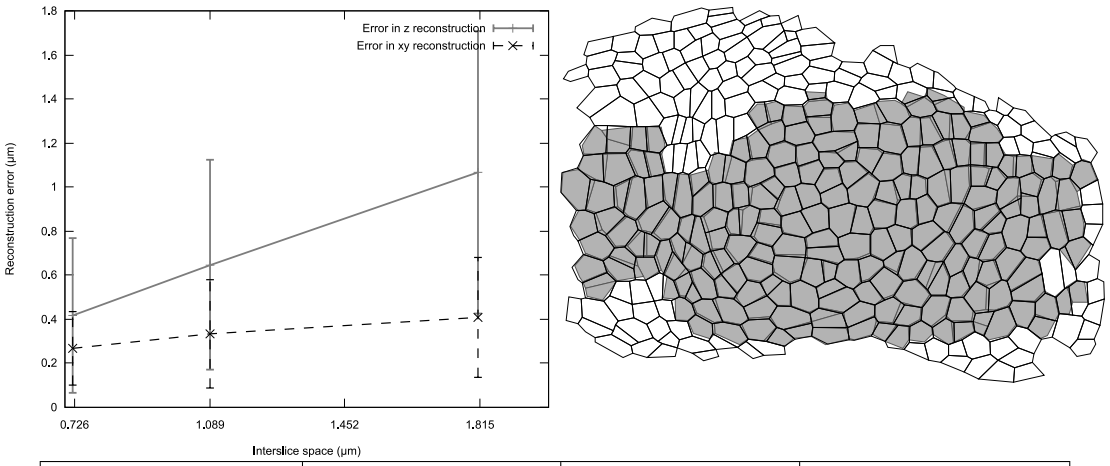
II.4.2 Conclusion

Different methods are currently available to analyse the dynamics of a three dimensional structure. Although the general approach is always the same, there does not seem to be a single standardised method available that is immediately applicable to all objects. As a consequence, the existing algorithms have to be adapted to the biological system that is used. Here we have described a full protocol to analyse the growth and development of the shoot apical meristem. The method is robust and gives access to a very wide range of quantitative data. In more general terms, the type of analysis described here will undoubtedly become more and more important in the future.

II.5 Material and methods

II.5.1 Plant culture and confocal microscopy

The plants were grown and observed essentially as described by Grandjean *et al.* (2004). Seeds were sown on petri-dishes with a medium adapted for Arabidopsis (Hamant *et al.*, 2002). After 2 days at 4℃, the seeds on medium were put in growth chambers at 20℃ and 16h of light. For NPA treatment, 10-5 to 10-6M of NPA was added to the medium. As soon as naked inflorescences had formed, the plants were transferred to medium without inhibitor. As soon as these inflorescences started to generate new flower buds they were examined with a Leica TCS-NT confocal laser scanning microscope (Leica, Heidelberg, FRG) with an argon/krypton laser (Omnichrome, Chino, CA, USA) and a AOTF for excitation. For this purpose, the meristems were embedded in a layer of low melting point agarose (Sigma, St Louis, USA), the tip of the meristem being close to the bottom of a WillCo-dish (WillCo Wells, Amsterdam, The Netherlands) with a glass bottom. Best results were obtained when the meristems were embedded while the agarose was not yet completely solid. The red vital dye FM4-64 (Molecular Probes Europe, Leiden, The Netherlands) was used to visualise the cells. Medium scan (450 lines per second) images (512×512 pixels) were generated using a long distance Leica 40×0.8NA water HCX APO



Inter-slices space	Surface coverage	Missed vertices	Phantom vertices
$0.726\mu m$	93.6%	75	18
$1.089\mu m$	91.6%	85	50
$1.815\mu m$	63.6%	283	19

FIG. II.7: **Optimal amount of sections for reconstruction.** The top left graphic represents the reconstruction error (i.e. the distance between the vertices in the sub-optimal reconstruction and the optimal one) in the z coordinate and in the (x, y) coordinates. These two kinds of error were separated because they come from different reconstruction steps. The error made seems a linear function of the distance between the sections. The graphic shows the mean values (linked by a line) and the standard deviations.
The top right image represents an optimal reconstruction (white) and a reconstruction made with sections at a distance of $1.815\mu m$ (in grey). Note that even at $1.815\mu m$ an important part of the meristem can be studied.
The table presents quantification of structural errors (i.e. missing or added vertices in the sub-optimal reconstructions) and the surface coverage for each tested intersection space. Note that the optimal reconstruction includes 722 vertices.

L. The inflorescences were observed in an inverted microscope DM IRB (Leica, Heidelberg, FRG). As a consequence, the plants were observed with their meristem down. As soon as a stack of images was obtained on a particular meristem, the plant was put back in the growth chamber, meristem up, to recover.

II.5.2 Quantification of errors.

Microscopy errors. Potential optical aberrations due to problems with the lenses or with the scanning devices of the microscope can lead to important imprecision in the quantification of cell size and shape. Since they cannot be evaluated on theoretical basis, they have to be estimated experimentally. For this purpose, we replaced the meristems by calibrated fluorescent spheres with a diameter $84\mu m$. To distinguish between the errors due to the lens and other factors, we also measured the spheres with another lens. This showed that our equipment did not lead to significant distortions.

Digitising errors. To evaluate the quality of the sub-optimal reconstructions, we compared them to the optimal reconstruction using three different criteria: (i) the surface of the meristem covered by the representation, (ii) the number of missed and added ('phantom') vertices when compared to the optimal reconstruction. Finally, (iii) the mean distance between corresponding vertices in the optimal and the suboptimal reconstructions were calculated. The values computed for these criteria are given in Figure II.7.

II.5.3 Implementation

Only a very general description of the protocol and algorithms will be given here. The process implied two different pieces of software developed especially for this purpose on GNU/Linux. The first one is called Merryproj and is used to compute the projection of the surface of the meristem. The second one is called Merrysim and is used to reconstruct the 3D-meristem from the projection and perform further analysis.

The user interface of Merryproj was created using Python and the PyQt graphical toolkit, but the algorithms were implemented in C++ and used to extend Python using the Boost.Python library (`http://www.boost.org/libs/python/index.html`).

Merrysim was mainly created using C++ and Qt and embeds a Python interpreter to perform analyses.

Both softwares are distributed under the terms of the GPL licence (`http://www.gnu.org/licenses/gpl.html`) and are freely available at `ftp://ftp.cirad.fr/pub/amap/VirtualPlants/merrysim/`. Moreover, all the details needed to implement the algorithms are described in the supplementary materials.

II.5.4 Cell lineage algorithm

The algorithm starts from a single cell and tries to identify its neighbours in two consecutive reconstructions of a meristem taken at time points t and t' ($t' > t$) as presented in Figure II.8. At the each step, the algorithm needs the association of one cell C and one of its vertices at time t, with the corresponding cell C' and one of its vertices at time t' (at the first step it is done manually). In case the cell C divided between t and t', C' corresponds to the merged daughter cells of C (see Supplementary material). This implies that the new vertices formed during division of C are ignored until the end of this step.

The algorithm next associates every vertex of C with the corresponding vertex of C'. This is achieved by the following manner. If there was no cell division of the neighbouring cells, the mapping of the vertices of C on the vertices of C' is straightforward. Otherwise, the algorithm localises in C' resulting from the divisions. This is illustrated in Figure II.8. In this figure, there is one more vertex in C' than in C. To find out which vertex is new in C', all the polygons created from C' by removing one vertex are compared to C. The orange polygon in the figure is the one closest to C and the algorithm concludes that v'_7 is the new vertex. To associate the vertices of C with the ones of C', the algorithm starts from v_1 which is known to be associated with v'_1 and follows C and the orange polygon in the same way, associating the vertices $v_{2\cdots6}$ with the vertices $v'_{2\cdots6}$ excluding v'_7. The neighbours of C are then associated with the neighbours of C' using the association of the vertices (see Supplementary material). This then allows the algorithm to map the vertices of C onto the corresponding ones of C'.

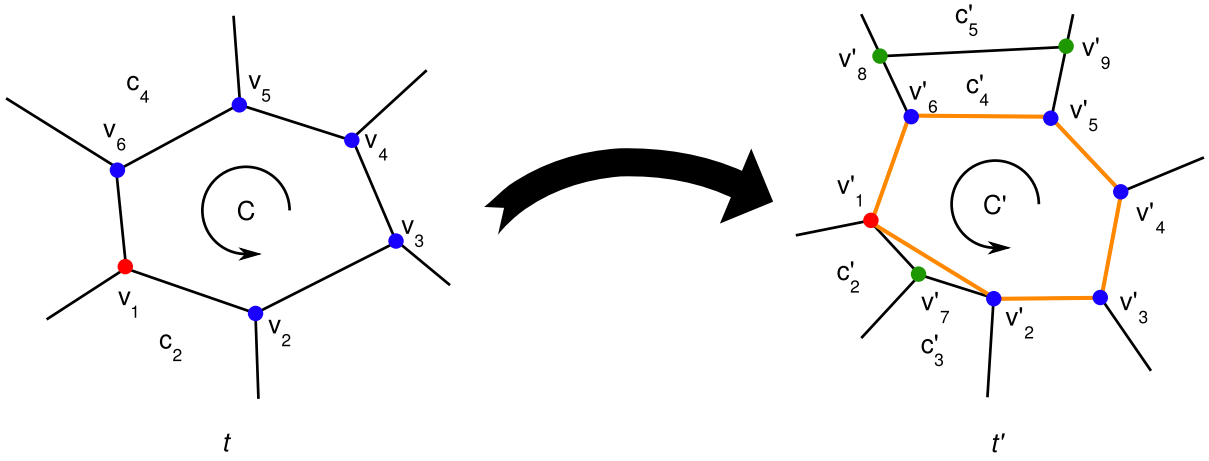


FIG. II.8: **Identification of neighbouring cells.** C and C' correspond to the same cell in two successive representations of the same meristem at time points t and t'. v_1 and v'_1 correspond to the same vertex at the two time points. From this knowledge, the algorithm tries to associate the neighbours of C with the ones of C'. The association of the vertices of C with the ones of C' is enough to associate the neighbours so that each neighbour c_n of C is associated with a neighbour c'_m of C' if and only if c'_m is either the same cell as c_n or a daughter-cell of c_n. In this figure, there is one more vertex in C' than in C. To find out which vertex is new in C', all the polygons created from C' by removing one vertex are compared to C. The orange polygon in the figure is the one closest to C and the algorithm concludes that v'_7 is the new vertex.

The mapping of the vertices directly allows the association of the edges of the polygons representing C and C'. Since every edge also belongs to one and only one neighbouring cell's polygon, it is possible to associate the neighbouring cells of C with the neighbouring cells of C'. For each neighbour of C three cases can be distinguished: (1) the neighbouring cell has not divided, (2) the neighbouring cell has divided such that a new vertex has formed in C' (see Figure II.8 where c_2 has divided into c'_2 and c'_3), (3) the neighbouring cell has divided but no new vertex has formed in C' (Figure II.8 where c_4 has divided into c'_4 and c'_5).

In the first case the algorithm simply associates the neighbouring cell of C with the one of C'. In the second case, the algorithm detects the division thanks to the presence of the new vertex. As a result the two daughter cells at t' are associated with their mother cell at t (c_2 in Figure II.8). However in case 3 no new vertex has formed, and in our example only c'_4 will be associated with c_4, and not c'_5. As a consequence, in contrast to the two other cases, this association is not complete as one of the daughter cell is missing. Since, based only on the association of local vertices the algorithm cannot distinguish between complete and incomplete cell associations, they all need further processing. To determine which ones are complete, the algorithm uses a heuristic: a confidence score is computed for each cell association, based on a comparison of their shapes (see Supplementary material), which will be very different if a mother cell is compared to only one of its daughter cells. The associations and their confidence scores are subsequently added to a global list containing all undetermined associations. The heuristic then considers the association of the list with

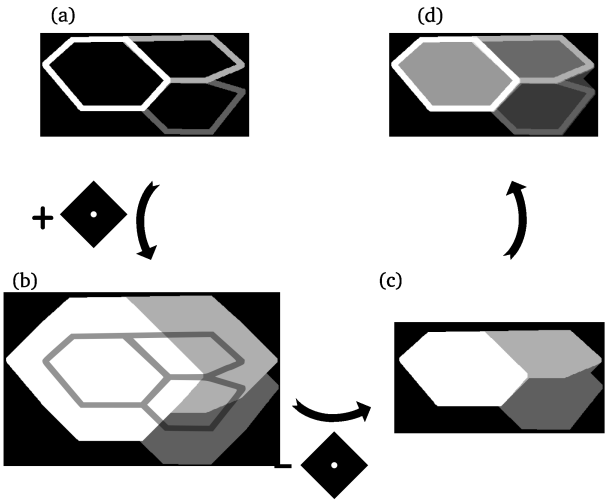


FIG. II.9: **Topological closure principle.** The topological closure of a grey scale image (a) by a structuring element S (the black square with the white dot) is the combination of two simpler operations: a topological growth and a topological erosion. A structuring element, in this case a square, is positioned relative to a reference point (here represented as a white dot). S can be placed anywhere on the image by means of the reference point. Consider now a point p on which the reference point of the structuring square S is placed. The topological growth (b) will set the intensity of p to the maximum intensity found in the area covered by S. After having applied this procedure to all pixels, a topological erosion is carried out (c). This is the symmetric operation (i.e. setting the intensity of p to the minimum intensity found in the area covered by S). In (d), the topological closure is superimposed on the original image for comparison.

highest confidence score as complete. It is removed from the list and the algorithm starts the whole procedure again from the selected associated cells, until the list of undetermined associations is empty.

II.6 Technical details

II.6.1 Transparency

The transparency mask of an image defines the transparency of each pixel as a value ranging from 0 (transparent) to 1 (opaque). It is usually called alpha mask or alpha channel when it is included in the image itself (Porter et Duff (1984)).

The only operation involving transparency used in this article is the blending of an image with a transparency mask and a background image (completely opaque). This operation is done pixel per pixel and channel per channel[1]. If α is the value of the transparency of a pixel, v the value of the pixel in the image and v' the value of pixel in the background, then the pixel in the resulting image will have a value v'' defined by:

$$v'' = \alpha v + (1 - \alpha)v'$$

[1]an image usually contains 3 channels: red, green and blue, but only one in case of a grey-scale image

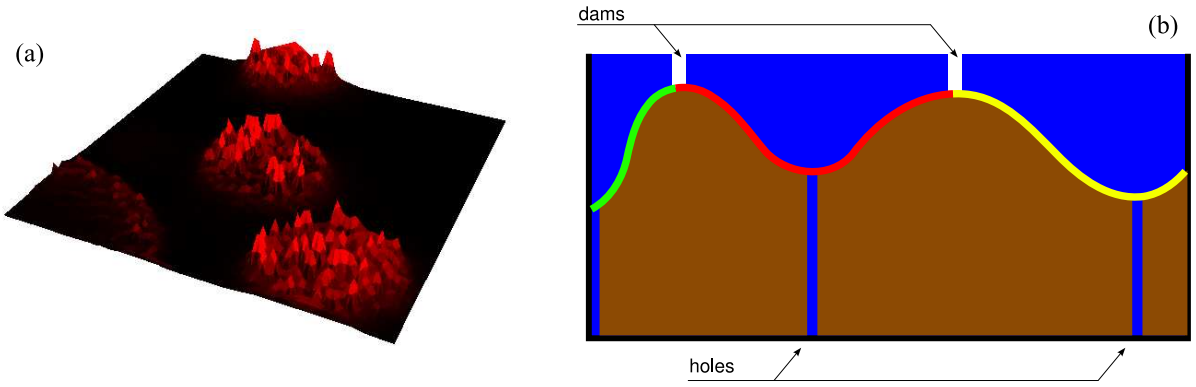


FIG. II.10: **Watershed principle.** Watersheds (WS) are used in grey-scale image segmentation to find objects and their borders. A greyscale image can be represented as a mountain map (a). In this map watersheds (catchment basins) can be detected by immersing the landscape step by step into water via holes in the minima of the image. When the waters caught by two different watersheds meet a dam is built to prevent them to join. These dams form then the watershed lines, which should correspond to cell boundaries (b).

Considering intermediate levels of transparency has two main advantages: first, it allows smooth transitions from one section to the next. Second, it lowers the impact of the dark regions on the final image, since they get a high transparency value. The final image luminosity is thus improved.

II.6.2 Computation of a reference system attached to the meristem

The position of the meristem in the reference system of the microscope changes between two reconstructions. To compute geometric or kinetic data at the meristem scale, the reconstruction of each observation of a given meristem should be made in a common reference system, attached to this meristem.

In the growing meristem, one cannot identify fixed points throughout time which could be used for defining such a reference system. Thus, we consider an arbitrary reference system for the reconstruction of the first meristem observation in time. This reference system will be used for all subsequent reconstructions of the meristem. To position these subsequent reconstructions in this reference system, we define a criterion based on minimal displacement of vertices. First, we assume that a central cell can be identified in each meristem reconstruction. Second, we assume that there exists a mapping associating vertices of at time t with vertices at time t' (see section: cell lineage identification). Third, we assume that between any two subsequent reconstructions this central cell does not move in the common reference system and there is no global rotation around it.

M and M' denote two subsequent meristems. Let $\mathcal{R}_0$ be their common reference system. Let $\{v_i, i = 1 \dots n\}$ and $\{v'_i, i = 1 \dots n\}$ be the sets of vertices of M and M' so that for each i, v_i is associated with v'_i. We suppose that the reconstruction of M is expressed in $\mathcal{R}_0$. We consider affine transforms $\mathcal{A}$ of M' preserving distances and oriented angles.

Our goal is to find $\mathcal{A}^*$, such that $M'' = \mathcal{A}^*(M')$ is close to M. More precisely, we want to find $\mathcal{A}^*$ such that the distance between M'' and M is minimal:

$$\mathcal{A}^* = \arg\min_{\mathcal{A}} E_{\mathcal{A}}(M, M') \tag{II.1}$$

where:

$$E_{\mathcal{A}}(M, M') = \sum_{i=1}^{n} e_{\mathcal{A}}(i)^2$$

with $e_{\mathcal{A}}(i)$ the local error for the associated vertices v_i and v'_i:

$$e_{\mathcal{A}}(i) = \|v_i - \mathcal{A}(v'_i)\| \tag{II.2}$$

where $e_{\mathcal{A}}(i)$ defines the distance between a vertex v_i in M and the transform of its associated vertex v'_i in M'.

To ensure that the identified central cells, C and C', are at the same position in $\mathcal{R}_0$, we add to the following condition on $\mathcal{A}^*$ to equation II.1:

$$\mathcal{A}^*(C') \approx C$$

To respect this constraint, we modify the definition of the local error (equation II.2):

$$e_{\mathcal{A}}(i) = \frac{1}{\|v_i - b_C\|}\|v_i - \mathcal{A}(v'_i)\| \tag{II.3}$$

where b_c is the centre of gravity of the central cell c.

This increases the contribution in the global error of the local errors on the central cell vertices. This thus ensures small displacement of the central cell, ideally corresponding only to the growth of the cell itself. Besides, this definition compensates also for the large displacements of vertices as they get farther from the central cell.

Numerical resolution Transformations in 3D space are described using homogeneous coordinate systems. This allows the expression of all kind of affine transforms uniformly, including translations. In a 3D homogeneous coordinate systems, points are defines by 4D vectors $V = [x_h, y_h, z_h, h]$ corresponding to 3D vectors $v = [x_h/h, y_h/h, z_h/h]$ if $h \neq 0$. In this coordinate system, every affine transform can be described as a 4x4 matrix. A linear application $\mathcal{A}$ preserving distances and oriented angles can be defined by the composition of a translation of vector $\vec{T} = [T_x, T_y, T_z]$ and a rotation, defined by the angle vector $\vec{\mathcal{R}} = (\alpha, \beta, \gamma)$. It can thus be described by a matrix $A_{\vec{T},\vec{\mathcal{R}}}$:

$$A_{\vec{T},\vec{\mathcal{R}}} = \begin{bmatrix} \cos\beta\cos\alpha & \cos\beta\sin\alpha & -\sin\beta & T_x \\ \sin\gamma\sin\beta\cos\alpha - \cos\gamma\sin\alpha & \cos\gamma\cos\alpha + \sin\gamma\sin\beta\sin\alpha & \sin\gamma\cos\beta & T_y \\ \sin\gamma\sin\alpha + \cos\gamma\sin\beta\cos\alpha & \cos\gamma\sin\beta\sin\alpha - \sin\gamma\cos\alpha & \cos\gamma\cos\beta & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The image V'' of a vertex V' by the transformation $A_{\vec{T},\vec{\mathcal{R}}}$ is given by :

$$V'' = A_{\vec{T},\vec{\mathcal{R}}} V'$$

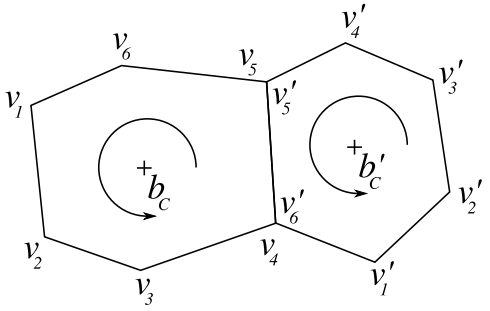


FIG. II.11: **Properties of vertices ordering for polygons.**

Equation II.3 can be rewritten as a matrix expression:

$$e_{A_{\vec{T},\vec{\mathcal{R}}}}(i) = \frac{1}{\|V_i - B_C\|}\|V_i - A_{\vec{T},\vec{\mathcal{R}}}V'_i\|$$

To solve the minimisation problem (equation II.1), we used an approximation method based on a min-square algorithm (see Marquardt (1963))[2].

II.6.3 Computational details of the cell lineage algorithm

Notations The next algorithms will use a common set of notations and conventions (see figure II.11).

- They all process two reconstructions M and M' of the same meristem at time points t and t' with $t < t'$.
- Each meristem reconstruction is described as a graph whose vertices are the cells and the edges the neighbourhood relationship between cells.
- Each cell C is described by a polygon p and a position b_C. The polygon represents the geometry of the cell and the position is the centre of gravity of this polygon.
- The polygons are star-shaped around their centre of gravity.
- Each polygon p is described as an ordered list of n vertices denoted $\{v_i\}_{1\leq i\leq n}$.
- The ordered list of vertices describing the polygons is cyclic. Thus:
  - We more generally define: $v_{kn+i} = v_i \quad \forall k \in \mathbb{Z}\,, \forall i \in \{1, \cdots, n\}$
  - Two ordered list differing only by a rotation of their elements define the same polygon: for all $i \in \{2, \cdots, n\}$, $\{v_1, \cdots, v_n\}$ and $\{v_i, \cdots, v_n, v_1, \cdots v_{i-1}\}$ denote the same polygon.
- The list of vertices is ordered so that: $\forall i \in \{1, \cdots, n\} \quad \left(\overrightarrow{b_c v_i}, \overrightarrow{b_c v_{i+1}}\right) \geq 0$

  This is possible because the polygons are star-shaped around b_C.

This section presents the computational details of the cell lineage algorithm needed to implement it properly.

[2]We used the implementation proposed in SciPy (see Jones *et al.* (2001)), which is a wrapper around MINPACK's algorithms (see Moré *et al.* (1980)).

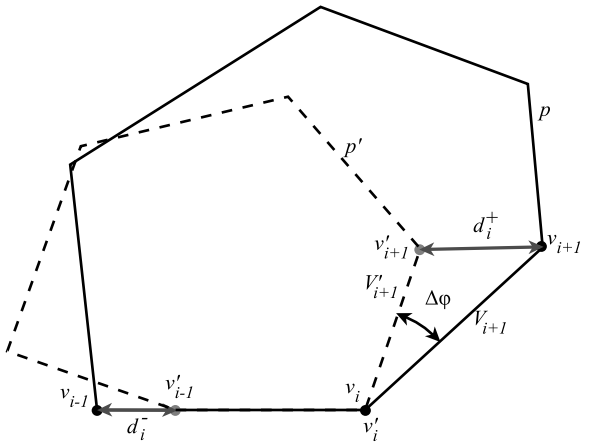


FIG. II.12: **Distortion measure for 4D algorithm.** This figure represents the measure of the local distance between the polygon p (in plain line) and p' (in dash line) at point i. In the figure, polygon p' was rotated so that $[v_i v_{i-1}]$ and $[v'_i v'_{i-1}]$ are aligned and translated so that v_i and v'_i are at the same position. The local distance is given by: $d_i = d_i^+ + d_i^-$

Confidence score The confidence score S of an association of cells is defined as the inverse of a metric d on the sets of polygons[3]:

$$S(C, C') = 1/d(p, p')$$

with $S(C, C') = 0$ if $d(p, p') = \infty$ and $S(C, C') = \infty$ if $d(p, p') = 0$.

We want the distance to be independent of the relative position of the polygons and of a possible rotation between them. It should also reflect the "intuitive" notion of difference between two polygons. Thus, the confidence score of the association of a cell C with a cell C', with their geometry being respectively represented by the polygons p and p', is defined as:

Therefore, if two polygons p and p' have exactly the same shape $d(p, p') = 0$, even if their position and orientation are different.

If the polygons have not the same number of vertices, we set the distance to infinity (in this algorithm, cell comparison is meaningful only when they do have the same number of vertices).

The distance d is computed between the polygons p and p' whose vertices are respectively $\{v_i\}_{1 \leq i \leq n}$, and $\{v'_i\}_{1 \leq i \leq n'}$. The ordered lists are rotated so that each vertex v_i of polygon p is associated with vertex v'_i of polygon p'.

For each polygon p, let us call k_i, the corner of polygon p at point i, $k_i = (v_{i-1}, v_i, v_{i+1})$. We define a local distance between p and p' at point i, $d_i(p, p')$, quantifying the difference between corner k_i in polygon p and k'_i in polygon p'. This difference will be expressed as the sum of two terms, corresponding to the edge difference before and after the point i. Formally, the local distance $d_i(p, p')$ is defined by:

$$d_i(p, p') = d_i^-(p, p') + d_i^+(p, p')$$

[3]In this paragraph, no particular assumption is made on the nature of the polygons

where

$$\begin{aligned} d_i^-(p,p') &= |V_i - V_i'| \\ d_i^+(p,p') &= \sqrt{{V_{i+1}}^2 + {V'_{i+1}}^2 - 2V_{i+1}V'_{i+1}\cos((\widehat{\overrightarrow{V_i}, \overrightarrow{V_{i+1}}}) - (\widehat{\overrightarrow{V_i'}, \overrightarrow{V_{i+1}'}}))} \end{aligned}$$

with $\overrightarrow{V_i} = \overrightarrow{v_{i-1}v_i}$, $\overrightarrow{V_i'} = \overrightarrow{v'_{i-1}v'_i}$, and $V_i = \|\overrightarrow{V_i}\|$.

Note that if two polygons have identical corners at point i, $d_i(p,p') = 0$. The geometric interpretation of this local distance is given in Figure II.12.

Then, the distance between p and p' is the average distance between all the corners of the polygons:

$$d(p,p') = \frac{1}{n}\sum_{i=1}^{n} d_i(p,p')$$

Identification of new vertices in one cell. In case of cell division in the neighbour cells of a cell C, new vertices can appear on the polygon defining the geometry of C. The cell lineage algorithm attempts to identify these new vertices in order to detect the cell divisions. The identification of new vertices is done for one cell C at time t associated with a cell C' at time t'.

Let us first assume that one vertex v_1 of the cell C is already associated with one vertex v'_1 of the cell C'. Let n be the number of vertices of C and n' the number of vertices of C'. Depending on $\Delta = n' - n$, the algorithm distinguishes three cases:

(i) $\Delta > 0$: There are Δ new vertices in cell C'

(ii) $\Delta = 0$: There isn't any new vertex in cell C'

(iii) $\Delta < 0$: This should not happen and denotes either an error in the meristem acquisition (vertices cannot disappear) or in the previous cell associations, or, a wrong hypothesis made for the delicate handling of 4-connected vertices (see below).

In case (i), to identify these new vertices, the algorithm evaluates all the confidence scores (see above) between the cell C and the cell C' where Δ vertices are removed. Note that, removing vertices from a cell simply comes down to removing them from the ordered list defining the cell polygon. All the $\binom{n'}{\Delta}$ possible ways to remove these Δ vertices in C' are explored. In practical situations, since Δ is not greater than 3, this process takes a time proportional to ${n'}^3$. The algorithm relies on the assumption that the highest confidence score is obtained when the actual Δ new vertices are removed from C'.

In case (ii), there is no new vertex to identify.

In case (iii), the new vertex identification procedure stops and reports an error.

Let us now assume that m vertices of C are already associated with vertices of C' (Note that $m < n$ and $m < n'$). We denote $\{v_{i_k}\}_{1\leq k\leq m}$ the vertices of C associated with the vertices $\{v'_{j_k}\}_{1\leq k\leq m}$ so that v_{i_k} is associated with v'_{j_k} (Note that for all k we have $1 \leq i_k \leq n$ and $1 \leq j_k \leq n'$). For convenience, we choose the indices so that $1 = i_1 < \cdots < i_m < n+1$ and $1 = j_1 < \cdots < j_m < n' + 1$ and, to take into account the cycling nature of polygons, we define for all $l \in \mathbb{Z} : i_{k+lm} = i_k + ln$ and $j_{k+lm'} = i_k + ln'$ with $k \in \{1, \cdots, m\}$.

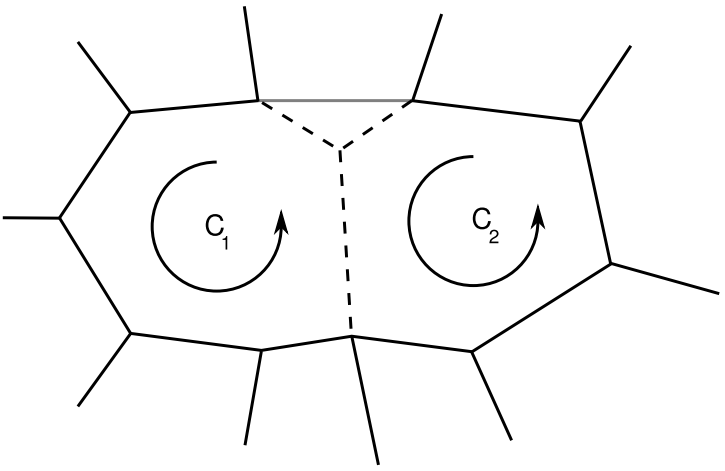


FIG. II.13: **Example of cells merging.** To merge cells C_1 and C_2, the algorithm removes the inner edges (dotted lines) and the 3-connected vertices of these edges. For each removed vertex, the algorithm links the vertex before in one cell with the vertex after of the other cell. As a result a closed polygon describing the merged cells is obtained. At the end, from a cell with 6 vertices and a cell with 7 vertices, the algorithm obtains a merged cell with 9 vertices.

In that case, the algorithm will look for new vertices between each $(v_{i_k}, v_{i_{k+1}})$ and not in the whole cell at once. Let us note $\delta_k = i_{k+1} - i_k$ and $\delta'_k = j_{k+1} - j_k$. Then we note $\Delta_k = \delta'_k - \delta_k$. There are new vertices between v'_{j_k} and $v'_{j_{k+1}}$ if and only if $\Delta_k > 0$. The existence of k so that $\Delta_k < 0$, is equivalent to the case (iii) (i.e. $\Delta < 0$) with the previous assumption. The actual new vertices are found as in the previous case, comparing polygons. However, this decreases the combinatorial complexity by limiting the number of possible positions for the position of the new vertices.

Handling of 4-connected vertices. In some cases, two vertices can be so close that they cannot be distinguished during acquisition. In that case, the reconstruction shows a vertex with four neighbour vertices, called a 4-connected vertex.

Until now, the described algorithm did not take into account the 4-connected vertices. These vertices may appear in three different configurations. First, two 3-connected vertices at time t can merge into a 4-connected vertex at time t'; second, one 4-connected vertex at time t can split into two 3-connected vertices at time t'; and third, a 4-connected vertex at time t can remain a 4-connected vertex at time t'. Therefore, a 4-connected vertex in a cell C (resp. C') can be associated with either one or two vertices in the cell C' (resp. C). When 4-connected vertices are encountered during the identification of new vertices (see above), it is *a priori* not possible to identify which of these two associations is the actual one. Both of them are thus systematically explored. The complexity of the algorithm is thus multiplied by 2^p, where p is the number of 4-connected vertices in both cells. Note that there are usually no more than 2-3 four-way vertices in the same cell and a limited of cells containing such vertices, leading to a reasonably bounded complexity.

Daughter cells merging When a cell at time t is associated with several cells at time t', the algorithm first merges the daughter cells to handle them as a single cell.

From the set of daughter cells, the merging process creates a new cell, deleting the inner edges and their vertices, except for the 4-connected vertices of the periphery that are always kept in the final cell (see Figure II.13).

The merging algorithm proceeds as follows:

input: the list of polygons[4] corresponding to the daughter cells and the set of 4-connected vertices.

output: the polygon corresponding to the merged cell.

The algorithm operates in 6 steps:

1. Create the set of all the edges of all the polygons in the form (v_i, v_{i+1}).
2. For each edge (v_i, v_j) for which (v_j, v_i) is also in the set of edges, mark (v_i, v_j), v_i and v_j for removal.
3. Remove all the marked edges.
4. Create a dictionary associating, for each remaining edge (v_i, v_j), v_i with v_j.
5. Create an empty polygon (i.e. an empty list of vertices).
6. Starting from a vertex from the perimeter not marked for removal, retrieve from the dictionary its associated vertex. Add this vertex to the polygon in construction if it is not marked for removal or if it is in the set of 4-connected vertices.

[4]Polygons are described an ordered lists of vertices such that any edge (v_i, v_j) of a polygon is the edge (v_j, v_i) of its neighbour.

Chapitre III

Modèle de transport d'auxine dans le méristème

III.1 Résumé de l'article

Le transport actif de l'auxine, une hormone végétale, joue un rôle majeur dans l'initiation des organes au niveau de l'apex caulinaire. Des protéines membranaire de la famille PIN facilitent ce transport (Vogler et Kuhlemeier, 2003; Galweiler *et al.*, 1998) et de récentes observations suggèrent que les maxima d'auxine créés par ces protéines sont à l'origine de l'initiation des organes (Reinhardt *et al.*, 2003b; Vernoux *et al.*, 2000). Cette hypothèse est basée sur une caractérisation qualitative purement visuelle de la répartition de la protéine PIN1 dans l'apex d'*Arabidopsis*. Aussi, pour pousser plus loins ces analyses, nous étudions les propriétés de cette répartition en utilisant les outils informatique pour la modélisation. Les simulations révèlent de nouvelles propriétés des distribution de PIN1. En particulier, elles suggèrent que le sommet du méristème joue un rôle important dans la distribution de l'auxine. Nous confirmons ces prédictions par de nouvelles expériences et proposons un modèle pour la dynamique des flux d'auxine dans l'apex caulinaire.

Les auteurs de l'article sont, dans l'ordre : Pierre Barbier de Reuille, Isabelle Bohn-Courseau, Karin Ljung, Halima Morin, Nicola Carraro, Christophe Godin et Jan Traas.

Ma participation concerne les aspects modèles et études numériques.

Cet article a été accepté à *Proceedings of the National Academy of Science of the USA* en Décembre 2005.

III.2 Introduction

There is strong evidence that active auxin transport, generated by influx and efflux carriers, creates patterns of auxin distribution at the shoot apex. This distribution is, in turn, interpreted in terms of differential growth and cell differentiation (Vernoux *et al.*, 2000; Reinhardt *et al.*, 2003b; Benkova *et al.*, 2003). In *Arabidopsis*, AUX1, a putative influx transporter (Bennett *et al.*, 1996), is mainly located in the surface layer (L1) of the shoot apical meristem (Figure III.1A ; Reinhardt *et al.*, 2003b). Since the protein seems to be homogeneously distributed in plasma-membranes of the individual cells, it has been

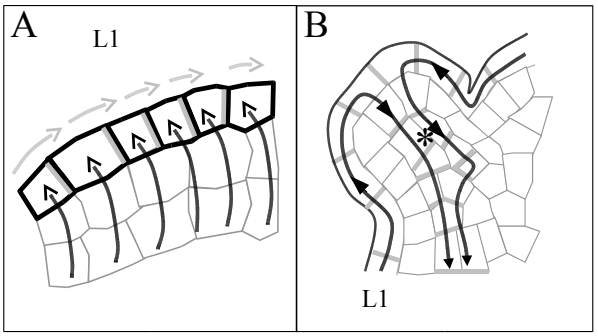


FIG. III.1: **Models for auxin transport in the shoot apical meristem.**
(A) The putative auxin influx carrier AUX1, represented in black, is homogeneously distributed on the cell membranes of the surface layer of the meristem, while the putative auxin efflux carrier PIN1 localization, represented in gray, seems to be polarized. As proposed by Reinhardt *et al.* (2003b), AUX1 would concentrate auxin in the surface layer (black arrows) and PIN1 would direct auxin fluxes (gray arrows) within these layers.
(B) In provascular tissues (*) in young primordia, PIN1 is oriented downwards, evacuating the auxin from the meristem surface (black arrows) to deeper tissues. Consequently, the primordia act as auxin sinks.

proposed that AUX1 restricts auxin to these layers. The efflux facilitator PIN1 is also localized in the surface layers of the meristem, but in contrast to AUX1 it is often localized on certain anticlinal sides of the cells only. Since neighboring cells often show coherent PIN1 positioning, it was proposed that PIN1 is responsible for directed hormone flows within the meristem L1 layer (Figure III.1A). In particular, careful immunological studies have revealed that the membranes carrying PIN1 are preferentially oriented towards the incipient primordia, suggesting auxin transport towards the young organs (Reinhardt *et al.*, 2003b; Benkova *et al.*, 2003).

Together the observations so far suggest a dynamic scenario where auxin is transported to the meristem from basally localized tissues via the L1 layer. In the meristem proper, auxin is redistributed and accumulates at particular sites where it will induce the initiation of new organs. This accumulation subsequently leads to the activation of transport in the provascular tissues causing an inward directed flow (Figure III.1B, our own non-published results). The young organ is thus transformed into an auxin sink, which depletes its surroundings from auxin and prevents the formation of new primordia in its vicinity.

Although this scenario is relatively straightforward, the previous observations leave a number of questions open. First, it is not clear at all, why auxin should start to accumulate at the site where a primordium will be initiated. Second, the immunolabelings reveal a very complex distribution of PIN1 proteins (Figure III.2). As a result the interpretation of these patterns in terms of cell-cell interaction networks and, more specifically, in terms of auxin distribution remains extremely difficult.

To address these questions, we developed computational modeling tools that allowed us to uncover novel aspects of the cell-cell interaction network and to predict auxin fluxes in the shoot apical meristems directly based on microscopical observations.

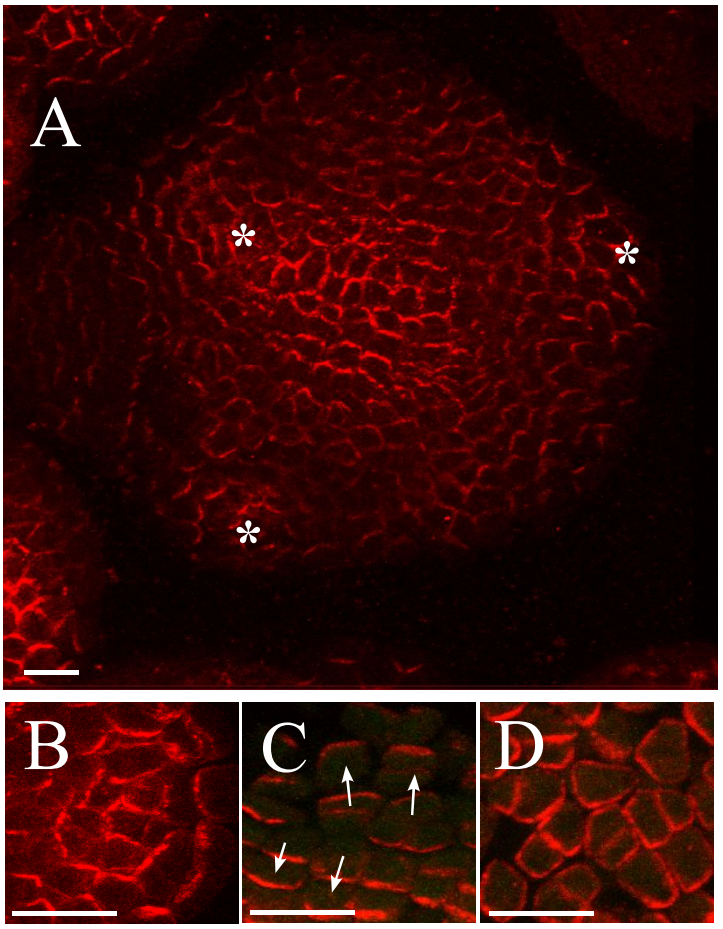


FIG. III.2: **PIN1 immunolocalization in *Arabidopsis* shoot apical meristems (Vitha *et al.*, 1997).**
(A) Global view of an anti-PIN1 immunolabeling on a meristem cross section. PIN1 is localized on the membrane and polarized in most cells. Patterns are complex. Bar, $20\mu m$.
(B) In the peripheral zone of the meristem, concentric PIN1 orientations around young primordia (asterisks) are observed. The patterns suggest that the cells orient towards a single central cell of the primordium.
(C) In boundaries between the meristem and the primordium, cell polarities in opposing directions are observed (arrows).
(D) At the meristem summit, PIN1 localization is variable and does not seem to show any particular organization. Bars, $10\mu m$.

III.3 Material and methods

III.3.1 Immunolabeling of PIN1 protein

After embedding, the meristems were sectioned perpendicular to the main stems with a thickness of $12-15\mu$m. After labeling with anti-PIN1, the physical sections were viewed in the confocal microscope to obtain an optimal image of the labeling patterns. In some cases, a single section was sufficient to cover the entire dome of the meristem. In other cases, the patterns of two successive sections were combined to cover the dome.

Anti-PIN1 Based on the sequence of *AtPIN1* (gene At1g73590), one potentially antigenic peptide sequence (GPTPRPSNYEEDGGPA) was selected in the large intra-cytosolic loop domain of AtPIN1 and used to produce antibodies (made by Eurogentec, Seraing, Belgium). This antibody recognizes PIN1, since no labeling is seen at the surface of the meristem in the *pin1* mutant. More detailed characterization of the antibody will be presented elsewhere. After immunostaining, the sections were viewed in a Leica confocal microscope to guarantee an optimal representation of the labeling patterns.

III.3.2 Gas chromatography and mass spectrometry (GCMS)

For GCMS, the plant tissue was collected in a 1.5 ml micro centrifuge tube and immediately frozen in liquid nitrogen. 0.5 ml cold 0.05 M phosphate buffer (pH 7.0) containing 0.02% sodium diethyldithiocarbamic acid (antioxidant) was added to the tube, together with $^{13}C_6$-IAA (Cambridge Isotope Laboratories, MA, USA) internal standard (50 pg/mg tissue) and a 3 mm tungsten-carbide bead. The sample was homogenized at 30 Hz in a vibration mill (Retsch MM 301, Haan, Germany) for 3 min, and then extracted under continuous shaking for 15 min at +4°C. After extraction, the pH was adjusted to 2.7 with 1 M HCl. Purification was performed using solid phase extraction on a 50 mg BondElut-C18 column (Varian, Middelburg, The Netherlands). The column was conditioned with 1 ml methanol, followed by 1 ml 1% acetic acid. After application of the sample, the column was washed with 1 ml 10% methanol in 1% acetic acid. The column was eluted with 1 ml methanol and the sample was then evaporated to dryness. 0.2 ml 2-propanol and 0.5 ml dichloromethane was added to the sample, followed by 5 μL 2 M trimethylsilyldiazomethane in hexane (Sigma-Aldrich, MO, USA). The sample was incubated in room temperature for 30 minutes, and excess diazomethane was then destroyed by adding 5 μL 2 M acetic acid in hexane. After evaporation to dryness, the sample was trimethylsilylated and analyzed by GC-SRM-MS as described.

III.3.3 Modeling tools

To interpret the labeling patterns in terms of putative auxin distribution, we developed a method relying on the simulation of auxin fluxes on digitized meristems.

Briefly, the method involves the following steps (Figure III.3A-G). First the membranes of the individual cells are identified on the images of immunolabeled sections. This information is used to reconstruct a graph where the nodes represent the cells and every cell is connected to its neighbors. These connections are used to simulate auxin diffusion

from cell to cell. A second type of connections is used to simulate active auxin transport. For this purpose, the cells are also connected via the membranes carrying PIN1 labeling. The latter connections are oriented (represented as arrows in figure III.3D-E) to take into account the direction of PIN1 mediated efflux. Using these maps of interconnected cells, we simulated auxin transport applying a set of rules based on observations and hypotheses mostly taken from the literature (for a detailed description see supporting information).

To test the robustness of the auxin distribution patterns, we performed a range of tests in which only one parameter was modified at the time (specified in supporting information). For each test, the non-varying parameters were set to values intermediate between those having extreme effects on the simulation. The results in particular showed that the patterns were qualitatively insensitive to major changes in diffusion and transport rates. At constant transport strength, the results were qualitatively equivalent for a thirteen-fold increase in diffusion rates. Conversely at constant diffusion rate the results were qualitatively equivalent for a fivefold increase in transport strength. The patterns should, therefore, be considered as robust.

In a minority of the cells, the immunolabeling was not clear enough to assert the polarity or even the presence of the PIN-protein. Therefore, we classified the different connections into four categories with decreasing confidence level: strong signal (i), strong but unpolarized signal (ii), weak but polarized signal (iii), and weak and unpolarized signal (iv). We next performed the simulations removing the connections ii - iv. As the resulting patterns were not significantly different, we only considered the labeled membranes with the highest confidence level (for details see supporting information).

III.4 Results

III.4.1 Simulation of auxin fluxes

The auxin transport through the network of interconnected cells is modeled using this set of hypotheses:

(i) Auxin passively diffuses via all membranes and is actively transported via oriented connections (Vogler et Kuhlemeier, 2003; Galweiler *et al.*, 1998; Reinhardt *et al.*, 2000, 2003b; Vernoux *et al.*, 2000; Benkova *et al.*, 2003; Bennett *et al.*, 1996).

(ii) Auxin is restricted to the L1 layer by AUX1 and enters the meristem from the meristem border via the efflux facilitator (Reinhardt *et al.*, 2003b).

(iii) Auxin is evacuated via the L1 cells that are in contact with provascular strands characterized by PIN1 labeling in deeper layers (Figure III.3G ; Vernoux *et al.*, 2000; Reinhardt *et al.*, 2003b). Longitudinal sections show that these provascular strands are about three cells wide (not shown). Therefore a circular area of three cells wide is designated to evacuate auxin at the position of each provascular strand on the images. They are defined here as 'Primordia' (P-1, P-2, ..., P-1 being the nearest to the meristem summit) and behave as auxin sinks.

(iv) The simulation algorithm continues to distribute the virtual auxin in the system until the auxin distribution gets stationary. This is to take into account that the establishment of auxin distribution is a fast process, much faster than growth and cell

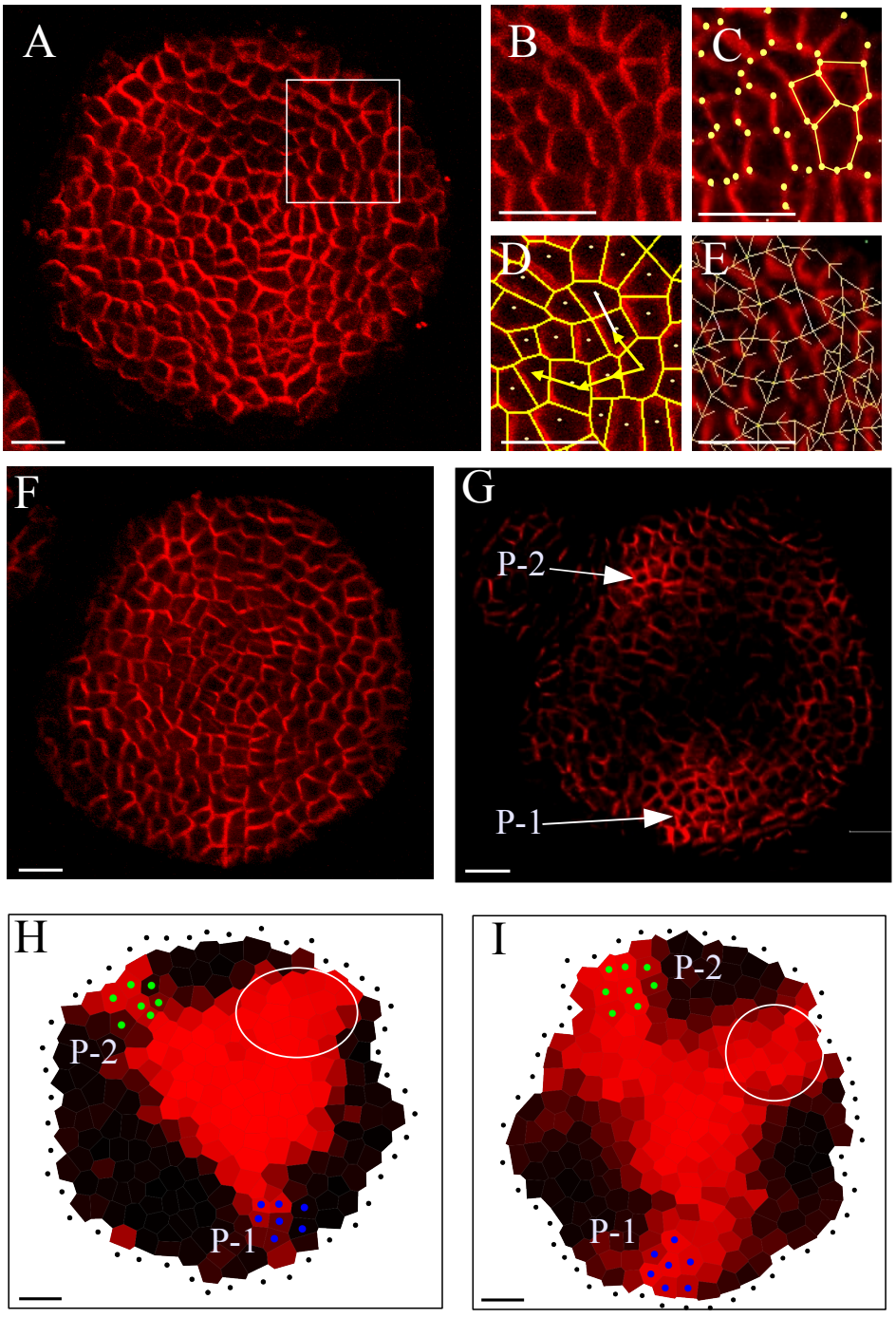


FIG. III.3

FIG. III.3: **From PIN1 immunolabeling to auxin fluxes simulation.**
(A) A transverse section showing PIN-labeling. The rectangle indicates the detail shown in **(B)**. Merrysim (see supporting information) is used to capture the cell shapes and the PIN1 localization in each cell.
(C) All cell vertices (spots) are manually positioned. The vertices of each cell are subsequently grouped.
(**D**) Cells are manually connected to each other if and only if there is a PIN1 labeling on the membrane between them (arrows). The connection is oriented in the way of supposed PIN1-mediated efflux.
(**E**) The result is a network of cell interactions.
(**F** and **G**) Anti-PIN1 immunolabeling on two successive transverse sections of another meristem. In (G), the labeling of the provascular strands at the level of P1 and P2 can be clearly distinguished. At these positions, called the primordium centers, auxin will be evacuated in the simulations.
(**H** and **I**). Results of the simulated auxin fluxes in meristems shown in (A) and (F). The position of the primordium centers visible on the original images are marked by green and blue dots. Virtual auxin is injected via the black dots surrounding the meristems. The quantity of virtual auxin per cell is proportional to the red intensity. Auxin accumulates where young primordia are being formed but also at the meristem summit. Moreover, the auxin maximum at the meristem summit protrudes toward the initium I-1 (gray circle).
Bars, $20\mu m$.

proliferation (Reinhardt *et al.*, 2003b). Therefore, in a normally growing meristem, auxin distribution is likely to be near the equilibrium at all times.

(v) Cells cannot accumulate auxin indefinitely. We modeled this constraint using a saturation level, above which the cells no longer accept auxin influx. Simulations tests showed that this was not very different from the situation where, at high level of accumulation, auxin diffusion overcomes active transport (see supporting information).

(vi) Auxin is degraded at a constant rate in each cell. Situations with different degradation levels were tested, including no degradation at all.

Ten meristems that were precisely sectioned in a plane transverse to the stem were immunolabeled (Vitha *et al.*, 1997) using a PIN antibody. Subsequently the corresponding images were used to extract connection maps. When the 6 rules mentioned above were applied to these maps (for technical details see supporting information), virtual auxin accumulated at the sites where young primordia were being formed (Figure III.3H-I). This property of the PIN1 network could be expected from the visual inspection of the immunolabelings. However, the simulations also showed a strong accumulation of virtual auxin in a domain covering the meristem summit, a property not obvious from visual inspection only. In all meristems tested, the central zone of accumulation also locally extended further to the periphery. Interestingly, this peripheral protrusion corresponded precisely to the site where the organ founder cells of the next primordium (called here initium-1 or I-1) were expected. Extensive tests shown that patterns were robust, relatively insensitive to even major changes in the parameters (see supporting information).

III.4.2 Auxin at the meristem summit

An unexpected simulation result was that the meristem summit accumulated auxin, suggesting a role for this domain in hormone distribution. Since previous studies only indicated a minor role for the meristem summit in this respect (Reinhardt *et al.*, 2003a), we next tested this prediction in planta. We first analyzed plants expressing *GFP* under control of *pDR5*, a synthetic promoter that is sensitive to auxin and that has been used to estimate relative hormone threshold levels in different tissues (Benkova *et al.*, 2003). As expected, GFP was strongly expressed in the future organ primordia, even at very early stages of initiation i.e. at the level of I-1, just next to the meristem summit (Figure III.4A; see also Benkova *et al.*, 2003). As a consequence, this pattern fully coincided with those predicted by the simulations. However, in contrast to what could be expected from the simulations, GFP was not, or very weakly, expressed in the meristem summit. Therefore, either the summit contained little or no auxin, or *pDR5* was insensitive to auxin in the meristem summit. To distinguish between these two possibilities, we treated young *in vitro* grown plants (Grandjean *et al.*, 2004) expressing *pDR5::GFP* with auxin in absence or presence of the auxin transport inhibitor NPA. The presence of 10^{-5}M auxin and 10^{-5}M NPA caused an important increase in the amount of *pDR5::GFP* expressing cells. However, the meristem summit never showed any increase in GFP activity, even in meristems where the entire periphery had activated the marker (Figure III.4B-E). We concluded that, as judged by *pDR5* activity, the central domain of the meristem was auxin-insensitive.

The observed insensitivity did not provide any information on the actual amount of auxin present in this domain. To address this issue, we used a monoclonal antibody directed against auxin to define local differences in auxin concentrations (Avsian-Kretchmer *et al.*, 2002, ; Monoclonal IAA antibody (Agdia, Elkart, Indiana, USA)). This showed a weak, but consistent labeling pattern, with an obvious maximum at the meristem summit (Figure III.4F-G).

To provide additional evidence that auxin did accumulate in the central part of the meristem, we extended our analysis to gas chromatography and mass spectrometry (GCMS, Ljung *et al.*, 2001; Edlund *et al.*, 1995). Since a normal wild-type meristem was too small to perform this type of analysis, we decided to use the *clavata3* (*clv3*) mutant. This mutant lacks a signaling peptide (CLV3) that is required to keep the central part of the meristem within certain size limits (Clark, 2001; Laux, 2003). If this peptide is absent, the central domain continues to grow, until it is several millimeters wide (Figure III.5A). To confirm that the central domain of the *clv3* meristem behaved like a normal wild-type meristem summit with regard to auxin sensitivity, we crossed the *pDR5::GFP* marker into the mutant background. In the enlarged dome of the mutant, we could only observe GFP fluorescence at the very periphery, close to the site of organ initiation (Figure III.5B-C). To determine whether the *clv3* summit did contain auxin or not, we next performed GCMS. For this purpose we measured the auxin contents in apices containing the SAM and young flower buds of *clv3* mutants. In addition, samples containing only cells coming from the enlarged meristematic summit of the *clv3* mutant were taken. The results (Figure III.5D) showed that the samples enriched in central zone cells contained active IAA, and were even enriched in hormone. Thus, the hypothesis that the central domain of the meristem is insensitive to auxin, but contains free IAA, as suggested by the computer simulations

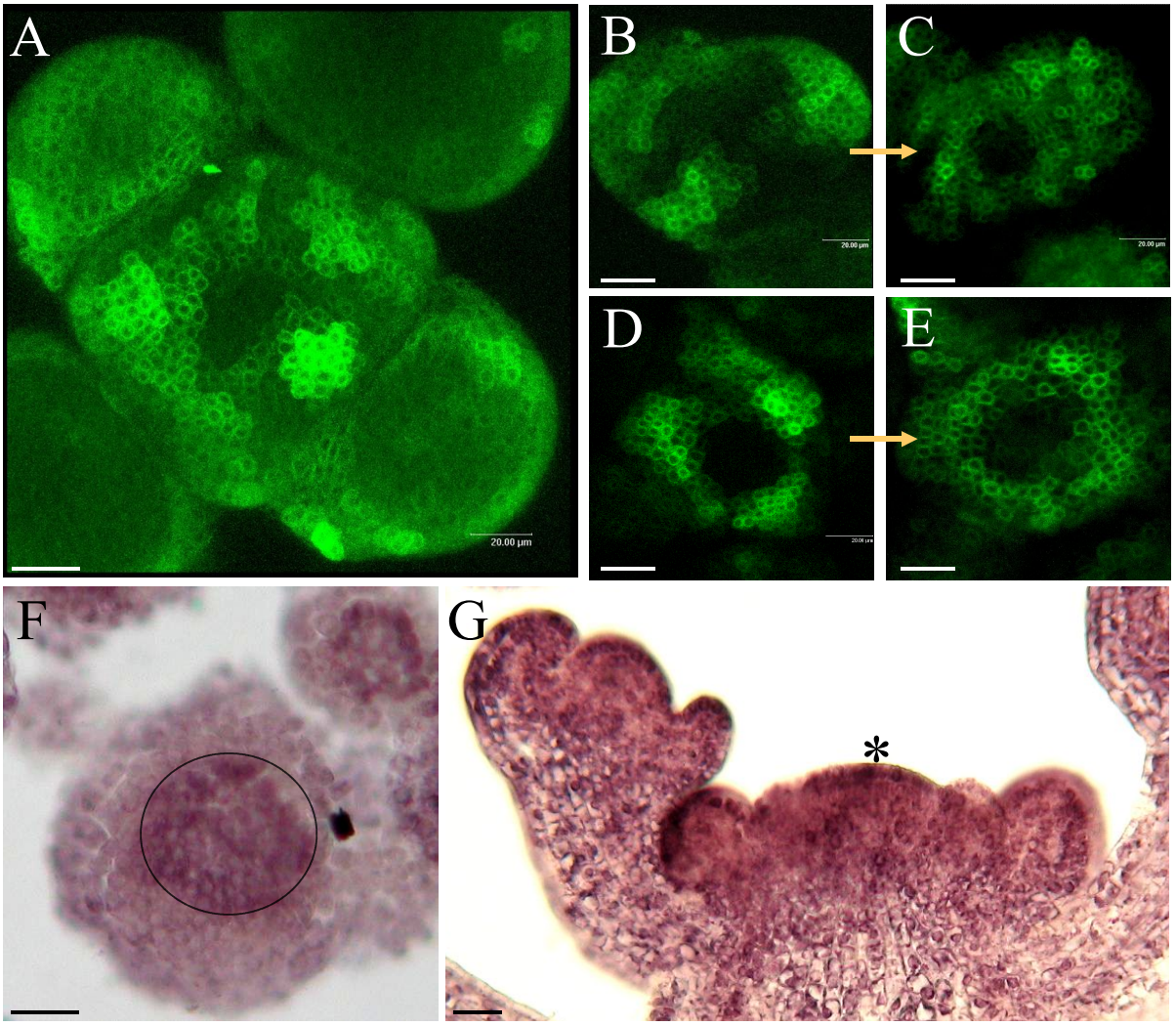


FIG. III.4: **Localization of auxin in *Arabidopsis* shoot apical meristems.**
(**A** to **E**) Spatial pattern of *pDR5::GFP* expression in shoot apical meristems under different conditions.
(A) Untreated meristem. (B and C) Treatment of a meristem with 10^{-5} M IAA during 22 hours. 10^{-5} M NPA (auxin transport inhibitor) was added to keep auxin in the meristem (B: t=0h, C: t=22h). (D and E) Treatment of a meristem with 10^{-5} M of the synthetic auxin 2,4 D during 22 hours (D: t=0h, E: t=22h). The *pDR5::GFP* expressing domain covers a larger part of the periphery after the treatment with IAA-NPA or 2,4 D but the summit of the meristem remains unlabeled.
(**F** and **G**) Immunolocalization of IAA in shoot apical meristems (Thomas *et al.*, 2002; Moctezuma et Lewis, 1999). The presence of labeling is characterized by a purple/brown signal.
(F) Cross section of a wild-type meristem; showing labeling at the meristem summit (arrow head). (G) Longitudinal section of a wild-type meristem also showing labeling at the meristem summit.
Bars, $20\mu m$.

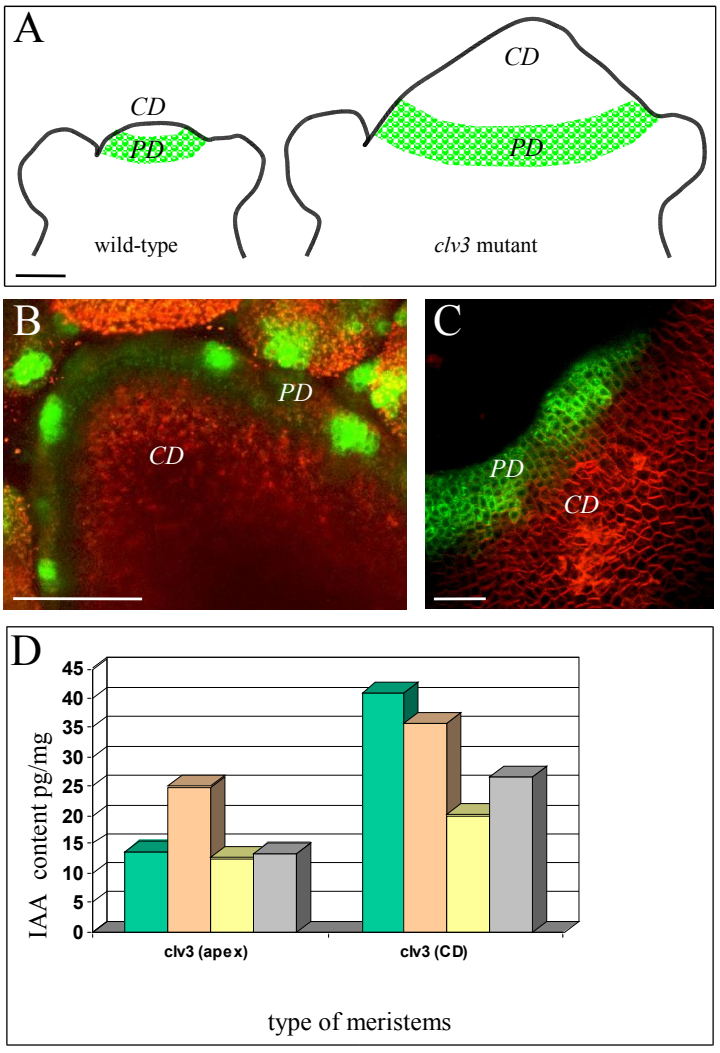


FIG. III.5: **Quantification of IAA in the central part of the *clv3* meristems.** (**A**) Schematic descriptions of wild-type and *clv3* meristems illustrating the enlarged central zone in *clv3* (CD: central domain). The green area represents the periphery domain (PD) where *pDR5::GFP* can be expressed.
(**B** and **C**) Pattern of *pDR5::GFP* expression in *clv3* meristems. (B) Global view of a full projection showing that *pDR5* activity is limited to the meristem periphery, with several maxima where the next primordia will be formed. (C) Detail of a meristem. Bars, $50\mu m$.
(**D**) Results of IAA quantification with GCMS in *clv3* meristems. Samples included the young apex (CD+PD+young primordia) or the central domain (CD) only. For each class, the quantification was performed on 4 different samples (4 colors), each sample containing several meristems. The quantification shows that the central domain of *clv3* meristems contains IAA.

and the auxin immunolabeling, was further confirmed using the IAA quantification in the *clv3* mutant.

III.4.3 Further simulation to test the role of auxin at the summit

What could be the function of IAA in the central domain of the meristem? To address this question, we performed additional simulations. These simulations were based on the same rules as before, but in addition the model was instructed to degrade auxin at the meristem summit. In all meristems tested, this additional instruction not only removed the auxin maximum from the meristem summit, but also the maximum at the level of the I-1 initium (Figure III.6A-B). By contrast, the maxima around the formed primordia were maintained. The results, therefore, suggest that the meristem summit plays an essential role in the creation of novel auxin maxima at the site of the organ primordium founder cells.

III.5 Discussion

Together, the simulations and subsequent experiments lead to a model, in which auxin coming from the periphery is transported into the central zone of the meristem. At a certain level of accumulation, auxin can no longer freely enter the meristem summit and because new auxin is arriving constantly, the hormone will accumulate at the site where the fluxes towards the summit are the most abundant. In a way, this would be analogous to a "traffic jam" at the entry of the meristem. Our simulations predict that this site corresponds precisely to the I-1 area, i.e. the zone where the inter-primordium distance is the largest (Figure III.7).

The results might seem in contradiction with elegant experiments where the tomato meristem summit was ablated using a laser (Reinhardt *et al.*, 2003a). In this case, no modification in organ positioning was observed, at least for a period of up to 4-5 plastochrones, suggesting that the meristem center did not play an important role in organ positioning. To clarify this issue we performed additional simulations, where all cells from the meristem center were removed (Figure III.6C). Interestingly, this did not have an effect on the accumulation of auxin at I-1 in the model. In this context, it should be noted that an ablated meristem center is analogous to a center which no longer accepts auxin. As a consequence, it would also cause an accumulation of auxin at the site where the fluxes are most abundant. Our results are, therefore fully compatible with the experimental evidence and provide an alternative explanation.

In conclusion, our results reveal a robust network of cell interactions which is sufficient to generate auxin distribution patterns consistent the observed organ positions (Lyndon, 1998). In addition they suggest a role for the meristem summit in organ positioning. The next, challenging step will now be to understand how the PIN1 proteins themselves are oriented. It is tempting to propose a model where the auxin gradients and fluxes themselves are at the basis of the patterns of cell polarity (Mitchison, 1981). At this stage, however, there is no evidence that the small gradients and fluxes that exist at the level of one cell will be sufficient to generate specific cell polarities. By any means, it will not only be important to identify cellular mechanisms leading to polar localization of PIN1, but we

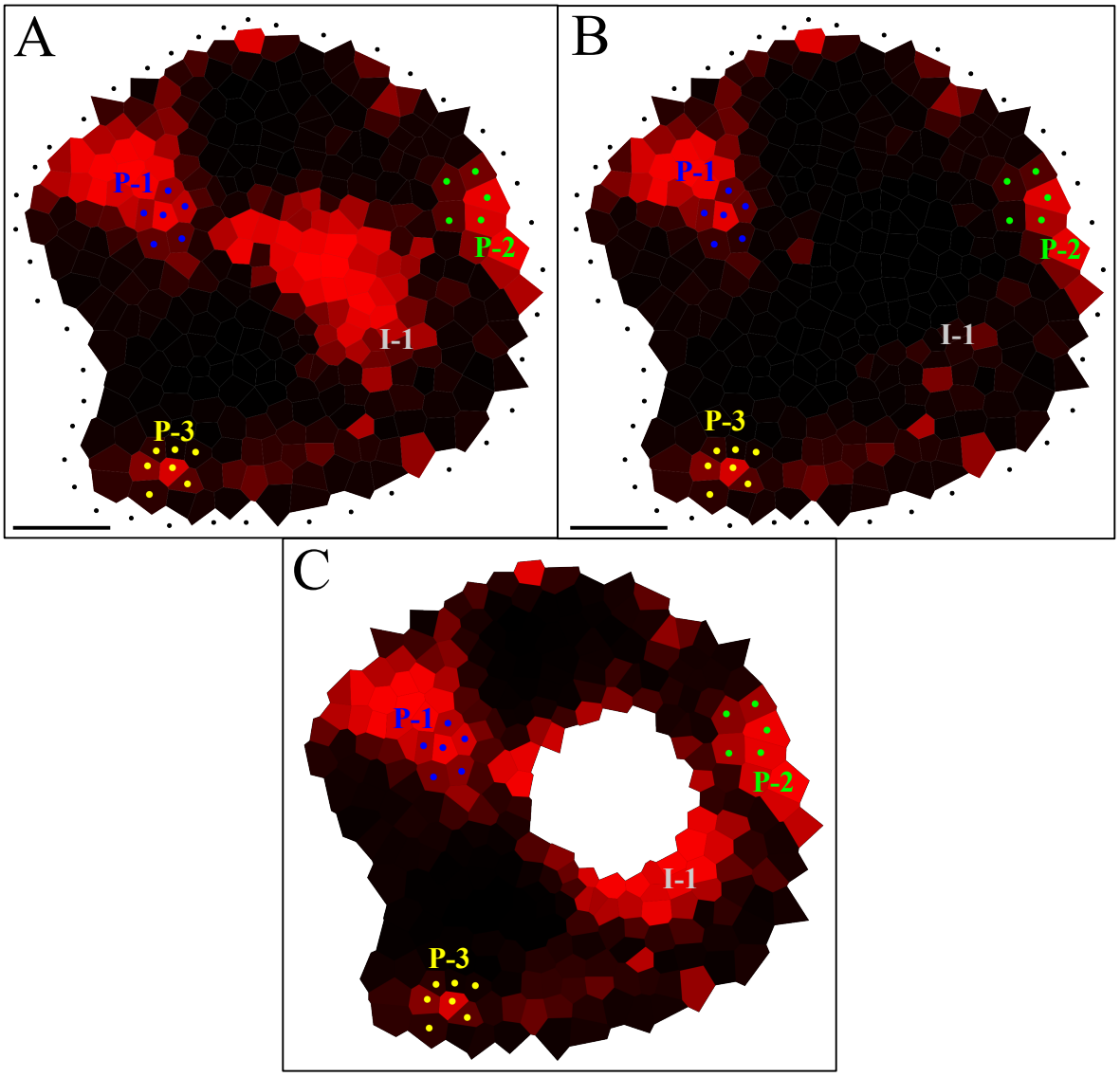


FIG. III.6: **Testing the importance of auxin accumulation at the meristem summit.**
(**A**) Simulation of auxin distribution using the standard parameter set (i.e. there are no special instructions for the meristem summit and auxin is evacuated only via the primordia P-1, P-2 and P-3).
(**B**) Simulation of auxin distribution in the same meristem, but this time the auxin arriving at the summit is immediately degraded. As a result, the maximum at the initium I-1 has disappeared.
(**C**) Simulation of auxin distribution in the same meristem, but this time, the meristem summit was removed. We defined this summit using the auxin accumulation zone. The initium I-1 is still present (A).

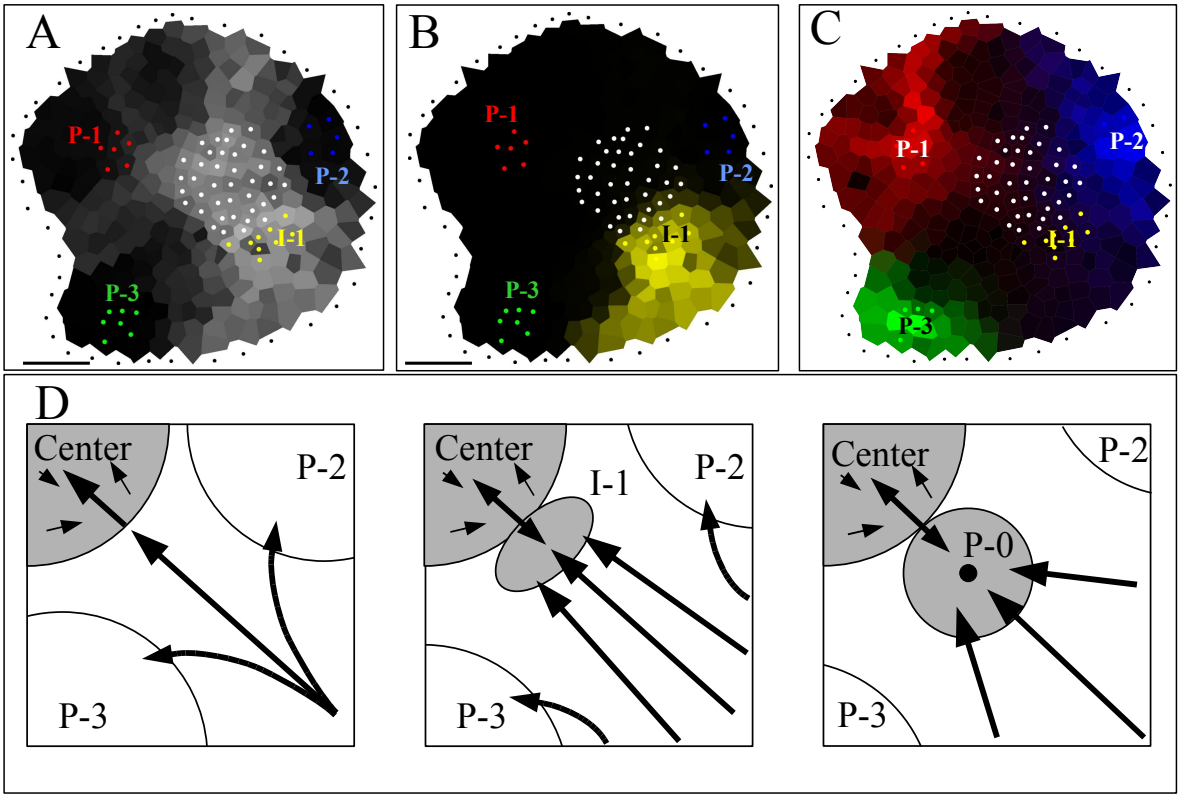


FIG. III.7: **Auxin fluxes and primordium initiation.**
(**A**, **B** and **C**) Auxin pathways inferred from a simulation (see supporting information). The color intensity in each cell is proportional to the contribution of this cell to auxin accumulation in the chosen zone (black: no contribution). The different zones are indicated as groups of colored dots.
(A) Auxin reaches the summit (gray dots) via corridors between primordia. The most important flux is between P-2 and P-3. I-1 is located at the limit of the summit and the most important flux towards the summit.
(B) The initium I-1 (yellow dots) is mainly filled by auxin coming from the periphery. PIN patterns suggest that the center contributes little.
(C) All three primordia receive auxin from the periphery. P-1 (red dots) and P-2 (blue dots) receive also some auxin from the center in contrast to P-3 (green dots).
(**D**) Model for the formation of an auxin maximum preceding creation of a primordium. As the distance between P-2 and P-3 increases, more auxin arrives at the meristem center in this sector. Since the center can only absorb a limited amount of auxin, this will lead to the formation of an auxin maximum (I-1). Eventually, this maximum will be transformed into a primordium (P-0) where the provascular system behaves as an auxin sink (black dot at the center of the primordium).
Bars, $20\mu m$.

need also to understand how these mechanisms are coordinated at the level of the whole meristem.

Acknowledgments. We would like to thank Dr. Przemyslaw Prusinkiewicz and Dr. Cris Kuhlemeier for critical reading of the text. We would also like to thank Dr. Jean-Louis Giavitto and Dr. Olivier Michel for providing the MGS language. We thank Dr. Giri Friml for providing the *DR5::GFP* line. PBdR was financed by an ASC fellowship provided by INRA. J.T. was financed by an ACI from the french ministry of research. I.B-C was financed by the french ministry of research

III.6 Technical details

III.6.1 Description of the model

A model was designed for simulating auxin transport in the meristem. The aim of this model is to study possible stationary regimes of auxin fluxes inferred from the observed distribution of auxin efflux facilitator PIN1.

Representation of the meristem The organization of the cells in the L1-layer is represented as an undirected graph, called the *topological graph*, in which vertices represent the cells and edges between vertices represent the cell walls. For each cell c in the topological graph, $\mathcal{N}(c)$ defines the set of cells adjacent to c. Similarly, the presence of the efflux facilitator protein PIN1 on cell walls, is represented by a second graph, called the *pump graph*, taking into account the orientation of the auxin transport. As in the case of the topological graph, vertices of the pump graph represent the meristem cells. However, edges are now oriented and denote the presence of PIN1 on cell membranes: there exists an oriented edge from cell c to cell n if PIN1 protein is present in c on the membrane next to n. We denote $\mathcal{P}_o(c)$ the set of neighbors of c reached by an outgoing edge from c and $\mathcal{P}_i(c)$ the set of neighbors of c that are the origin of an incoming edge in c.

The topological and pump graphs are constructed by hand using a dedicated software, called Merrysim, developed to ease the operation.

In some cases, the presence or orientation of PIN1 protein on cell walls is difficult to assert without ambiguity. For this reason, we considered four levels of confidence in the identification of the auxin pumps:

1. PIN1 is clearly seen with its polarity (89% of the links in all the meristems)
2. PIN1 is clearly seen but its polarity cannot be determined (4% of the links)
3. PIN1 presence and orientation are discerned with difficulty (6% of the links)
4. PIN1 presence is discerned with difficulty and its polarity cannot be determined (1% of the links)

Edges in the pump graph are generated depending on this level of confidence. For each pump at confidence level 1 or 3, a single oriented edge is created in the pump graph. For each pump at level 2 or 4, two oriented edges (in both directions) are created. These levels are recorded for each edge.

Each cell c is associated with a state defining its properties. This state includes the volume of the cell (V_c), the concentration of auxin in the cell (a_c), the presence of PIN1 proteins oriented towards the provascular system of the meristem (E_c where $E_c = 1$ if there is such PIN1 proteins, otherwise $E_c = 0$) and the relative efficiency of these pumps (β_c, see below). Of these properties, only the concentration of auxin is variable throughout time.

Transport process Auxin distribution is assumed to depend on a combination of four mechanisms: (i) passive diffusion of auxin through cell walls, (ii) active transport of auxin facilitated by PIN1 protein in the surface of the meristem, (iii) active transport of auxin toward the deeper layers of the meristem and (iv) natural turn-over of auxin.

(i) Diffusion is a passive process that tends toward the equilibrium of concentrations. We assume that the quantity of auxin diffusing through a cell wall is proportional to the difference of concentrations between the cells sharing the wall. Considering a cell c and one of its neighbors n, then:

$$\gamma_{c \leftarrow n}(t) = \Gamma \left[a_n(t) - a_c(t) \right] \tag{III.1}$$

where $\gamma_{c \leftarrow n}$ is the quantity of auxin per time unit diffused from n to c and Γ is the diffusion coefficient. Note that $\gamma_{c \leftarrow n}$ is positive if auxin enters cell c.

(ii) Active transport inside the meristem surface is modeled as a quasi-constant pumping of auxin. Indeed, if the auxin concentration is too low, pumps are less efficient in recruiting auxin molecules. Thus, PIN1 pumping efficiency decreases when the source cell has a very low amount of auxin. Symmetrically, we can suppose that a single cell cannot concentrate auxin more than a saturation threshold (σ). The efficiency of PIN1 pumping must then decrease when the destination cell approaches this limit, and reaches 0 when the cell is completely saturated. Simulations were performed with and without this hypothesis. When the source cell has enough auxin and the destination cell does not saturate, conditions are optimal for maximum efficiency of the pumps and the pumping is constant. The amount of auxin pumped by PIN1 protein is thus defined by:

$$\psi_{c \leftarrow n}(t) = \Psi \left(1 - e^{-\alpha a_n(t)}\right) \left(1 - e^{-\alpha(\sigma - a_c(t))}\right) \tag{III.2}$$

where $\psi_{c \leftarrow n}$ is the quantity of auxin per time unit pumped from n to c by the PIN1 protein in n on the wall toward c, Ψ is the maximum efficiency of the pumps, α is the slope of $\psi_{c \leftarrow n}(t)$ when either the n-auxin is depleted or when c is saturated. In this formula, the first parenthesis models the behavior of the pump when auxin in the source cell is depleted and the second parenthesis when the target cell is saturated. The product denotes independent handling of these two special cases.

(iii) In primordia cells, auxin is evacuated toward deeper layers by PIN1 proteins pumping auxin to the provascular internal cells. This pumping is similar to the pumping inside the L1-layer except that there is no saturation threshold in the inner layers:

$$\psi_{c\downarrow}(t) = \beta_c \Psi \left(1 - e^{-\alpha a_c(t)}\right) \tag{III.3}$$

where $\psi_{c\downarrow}$ is the quantity of auxin per time unit pumped from c toward the inner layers of the meristem by the PIN1 protein in c, Ψ is the maximum efficiency of the downward

pumps and β_c is a coefficient decreasing with the distance to the primordium center. Pumping is assumed to be maximum in the primordium center and to exist also in the other primordium cells, with a decreasing strength as the primordia cells are positioned farther from the central cell. This is to take into account the observed orientations and quantities of PIN1 labeling in the immunolabelings (Reinhardt *et al.*, 2003b; Traas).

(iv) The turn-over of auxin is modeled as auxin degradation:

$$\tau_c(t) = \delta a_c(t) \tag{III.4}$$

where τ_c is the variation of auxin concentration and δ the degradation factor.

The global variation of auxin within a cell c is thus defined by:

$$\frac{da_c}{dt}(t) = \frac{1}{V_c}\left(\sum_{n\in\mathcal{N}(c)} \gamma_{c\leftarrow n}(t) + \sum_{n\in\mathcal{P}_i(c)} \psi_{c\leftarrow n}(t) - \sum_{n\in\mathcal{P}_o(c)} \psi_{n\leftarrow c}(t) - E_c\psi_{c\downarrow}(t)\right) - \tau_c(t) \tag{III.5}$$

which can be expanded into:

$$\begin{aligned}\frac{da_c}{dt}(t) &= \sum_{n\in\mathcal{N}(c)} \frac{\Gamma}{V_c}\left(a_n(t) - a_c(t)\right)\\ &+ \sum_{n\in\mathcal{P}_i(c)} \frac{\Psi}{V_c}\left(1 - e^{-\alpha a_n(t)}\right)\left(1 - e^{-\alpha(\sigma - a_c(t))}\right)\\ &- \sum_{n\in\mathcal{P}_o(c)} \frac{\Psi}{V_c}\left(1 - e^{-\alpha a_c(t)}\right)\left(1 - e^{-\alpha(\sigma - a_n(t))}\right)\\ &- E_c\frac{\Psi}{V_c}\left(1 - e^{-\alpha a_c(t)}\right) - \delta a_c(t)\end{aligned} \tag{III.6}$$

Boundary conditions. As stated in the main text, auxin is supposed to arrive in the meristem from the lower part of the plant via the L1-layer.

To simulate this behavior, a ring of auxin injection points representing "the lower parts of the plant" is added during meristem digitizing. Each of these points is linked to each of its neighboring cells in the meristem by a pump. These pumps are ruled by an equation similar to equation (III.2). In particular, the external injection points always provide enough auxin to guarantee maximal pumping. If the injection points do not provide enough auxin, no auxin accumulations are observed in the meristem. If they provide too much, the meristem is almost completed saturated by auxin (see Figure III.8 and text below). Based on tests, we chose a value for auxin injection inbetween these two extremes. Thus, the equation ruling the quantity of auxin exported by the external cell n into the internal cell c is:

$$\psi_{c\leftarrow n}(t) = \Psi'\left(1 - e^{-\alpha(\sigma - a_c(t))}\right) \tag{III.7}$$

III.6.2 Implementation

A dedicated software, Merrysim, was developed to create the topological and the pump graphs, to carry out numerical simulations and to analyze the results. It is released under the terms of the GPL licence (GPL).

Data acquisition and analysis tools were implemented in C++ and Python using the Qt3 toolkit for the design of user interaction (`http://www.trolltech.com`).

A simulation engine was developed in MGS, a language dedicated to simulation of dynamic systems with dynamic structures (Giavitto et Michel, 2001a,b). To solve the system of ordinary differential equaltions (ODEs) we used an explicit Euler integration scheme with constant time step dt (Press *et al.*, 1992). To avoid stability problems, we used a first order approximation, when $a_c(t)$ vanishes to 0. This provides a condition for $a_c(t)$ not to become negative:

$$a_c(t) \geq -(N_{max}+1)\frac{\Psi}{V_{min}}\alpha a_c(t)dt \tag{III.8}$$

where N_{max} is the maximum number of neighbors of a cell in the meristem and V_{min} is the volume of the smallest cell. This condition enables us to define an upper bound for dt:

$$dt \leq \frac{V_{min}}{(N_{max}+1)\alpha\Psi} \tag{III.9}$$

Visualizations were made using the Python module SVGdraw (`http://www2.sfk.nl/svg`) to transform the simulation results into SVG files.

The different parts of the software system are coordinated using the Python language. Merrysim is a free software that can be downloaded freely at: `ftp://ftp.cirad.fr/pub/amap/VirtualPlants/merrysim/`

III.6.3 Sensitivity analysis

To express the biological knowledge of the processes involved in the above model, we introduced parameters (e.g. the saturation threshold, the degradation rate) whose actual values cannot be estimated directly from experiments or cannot be exactly determined because of limitations in the original images (e.g. presence or orientation of certain pumps).

In order to test the validity of the conclusions, we made a number of changes in the values of these parameters to study the effect of these changes on the overall results. The set of parameters involved in the model is reflected in equation III.6. These parameters can be separated into two groups: i) *the structural parameters* whose values are set during meristem digitization and have ordinal or symbolic values. This group includes: $\mathcal{N}(c)$, $\mathcal{P}_i(c)$ ($\mathcal{P}_o(c)$) and E_c. ii) *the functional parameters*: Γ, α, σ, δ, Ψ and Ψ' that have real values.

To test the sensitivity of the model to the different parameters, two different procedures were used depending on the group of the parameter tested.

For the first group of parameters, different configurations of $\mathcal{P}_i(c)$ and $\mathcal{P}_o(c)$ were tested, according to the confidence level of the identified PIN1 pumps. As these pumps were classified into four categories with decreasing confidence level, simulations were carried out using the n first categories with n varying from 1 to 4. This allowed us to perform simulations using first only pumps whose presence was certain (i.e. 90% of the pumps), and integrating progressively pumps with lower confidence level.

For the second group, we tested the variation of each parameter independently. Since auxin concentrations are entirely determined by the relative values of the parameters at a

	Value Index								
Parameter	0	1	2	3	4	5	6	7	8
Ψ' (auxin import rate)	0	0.48	0.93	1.8	3.5	6.8	13	26	50
δ (degradation)	0	0.0007	0.0015	0.003	0.006	0.012	0.025	0.05	0.1
Ψ (pumping strength)	0	5	8	13	20	25	30	35	40
Γ (diffusion)	0	0.004	0.008	0.016	0.03	0.057	0.11	0.21	0.4

TAB. III.1: **Values used for the sensitivity tests.** Note that the values tested follow almost always an exponential law.

scale factor close, we were able to fix the value of one parameter: in all the simulations, the saturation parameter was set to 255 (auxin concentration thus ranging from 0 to 255 in each cell). For the other parameters, we used a range of nine different values (depicted in the table III.1) and fixed the other parameters to the following reference values: $\Gamma = 0.03$, $\alpha = 0.025$, $\delta = 0.006$, $\Psi = 20.0$, $\Psi' = 3.5$.

III.6.3.1 Results

For all meristems, we observed the same general pattern of auxin accumulation (see main text). Auxin was accumulating in the primordia and at the meristem summit. In this latter zone, a protrusion could always be seen in the direction of the initium I-1. We subsequently carried out the complete set of robustness tests on 5 meristems.

The auxin distribution patterns obtained using reference values were used as a starting point for these tests.

Auxin injection rate (Ψ'). Figure III.8. As the auxin flux from the injection points to the meristem increases, the total amount of auxin in the meristem augments. With increasing hormone quantities, auxin starts to accumulate first at the meristem summit, followed by the primordia and the initium. When the injection rate increases, almost the entire meristem is filled with auxin. However, some of its parts remain at a low auxin concentration. The patterns are relatively sensitive to this parameter as they do not support more than a twofold variation.

Degradation factor (δ). Figure III.9. The degradation factor also affects the total quantity of auxin in the meristem. In this case the patterns are relatively insensitive as auxin accumulation can still be observed at the primordia sites at low total auxin quantities (i.e. high degradation rates). The patterns support a sixteen-fold variation.

Pumping strength (Ψ). Figure III.10. Without pumping, the meristem is filled by the periphery and auxin diffuses toward the center. For a non null pumping, typical auxin accumulation patterns are readily observed, even for the weakest pumping forces tested. With increasing pumping forces, a better separation of the center and the primordia can be observed, mainly due to a stronger counter-effect on diffusion. In this case, the patterns support a fivefold variation.

Parameter	Minimum	Maximum	Ratio (max/min)
Ψ'	1.8	3.5	2
δ	0.003	0.05	16
Ψ	8	40+	5+
Γ	0.008	0.11	13

TAB. III.2: **Ranges of each parameter for which the auxin patterns are stable.** Note that the variations on pumping factor shows a stable pattern even for the highest value tested. The actual range of stability may thus be even greater than the one presented here.

Diffusion rate (Γ). Figure III.11. Without diffusion, most of the cells are either saturated or empty and accumulation sites cannot always be clearly identified. With the increase of the diffusion rate, the accumulation zones are better formed and enlarge until auxin is almost regularly distributed in the meristem. In this case, the patterns support a thirteen-fold variation.

Saturation (σ). Figure III.12. Without any saturation threshold, the auxin accumulation patterns are not significantly changed. However, very high auxin concentration in the meristem summit are observed (much higher than in the primordia).

Simulations using pumps with varying confidence level. Figure III.13. No significant differences between simulations using all the pumps or only pumps with highest confidence level could be observed.

This sensitivity analysis showed that the auxin accumulation pattern predicted by the simulations was particularly robust to changes in almost all the model parameters. The actual ranges of parameter values for which the auxin accumulation pattern of each meristem could be identified are summarized in the table III.2.

III.6.4 Tracking auxin paths in the meristem

We also identified the paths followed by auxin molecules to reach different zones of the meristem. Let us denote $q_c(t)$ the total amount of auxin molecules contained in cell c at time t. We consider a cell c' different from c and we define $q_{c \leftarrow c'}(t)$ the quantity of auxin contained in c at time t that went through cell c' at some anterior date. This number of molecules divided by the cell volume V_c defines a concentration of auxin molecules in c at time t that went through c', denoted by $a_{c \leftarrow c'}(t)$. By extension, we define $a_{c \leftarrow c}(t) = a_c(t)$.

At each time step t, a cell c looses globally a quantity $f_c^-(t)$ of auxin (via pumping, diffusion and degradation) and receives from each of its neighbor n a quantity $f_{c \leftarrow n}^+(t)$ of auxin. To compute $q_{c \leftarrow c'}(t)$ when $c \neq c'$ we add the hypothesis that the auxin arriving in a cell merges instantaneously with the auxin already present in this cell. This implies that when a quantity $f_{c \leftarrow n}^+(t)$ of auxin goes from cell n to cell c, the variation of $q_{c \leftarrow c'}$ is determined by the product between the proportion of auxin in n that went through c' and

the total quantity of auxin going from n to c:

$$d^+q_{c\leftarrow c'}(t) = \sum_{n\in\mathcal{N}(c)} \frac{q_{n\leftarrow c'}(t)}{q_n(t)} f^+_{c\leftarrow n}(t)dt \tag{III.10}$$

Similarly, when a quantity $f^-_c(t)$ of auxin leaves the cell c, the variation of $q_{c\leftarrow c'}$ is determined by the product between the proportion of auxin in c that went through c' and the quantity of auxin going out of c:

$$d^-q_{c\leftarrow c'}(t) = \frac{q_{c\leftarrow c'}(t)}{q_c(t)} f^-_c(t)dt \tag{III.11}$$

The total variation of $q_{c\leftarrow c'}, c \neq c'$ is thus:

$$dq_{c\leftarrow c'}(t) = d^+q_{c\leftarrow c'}(t) + d^-q_{c\leftarrow c'}(t) \tag{III.12}$$

$$\frac{dq_{c\leftarrow c'}}{dt}(t) = \sum_{n\in\mathcal{N}(c)} \frac{q_{n\leftarrow c'}(t)}{q_n(t)} f^+_{c\leftarrow n}(t) - \frac{q_{c\leftarrow c'}(t)}{q_c(t)} f^-_c(t) \tag{III.13}$$

which leads to the following relationships on concentrations:

$$\frac{da_{c\leftarrow c'}}{dt}(t) = \begin{cases} \displaystyle\sum_{n\in\mathcal{N}(c)} \frac{a_{n\leftarrow c'}(t)}{a_n(t)V_c} f^+_{c\leftarrow n}(t) - \frac{a_{c\leftarrow c'}(t)}{a_c(t)V_c} f^-_c(t) & \text{if } c \neq c' \\ \dfrac{da_c}{dt}(t) & \text{otherwise} \end{cases} \tag{III.14}$$

where the two fluxes, $f^+_{c\leftarrow n}(t)$ and $f^-_c(t)$ are defined by:

$$f^+_{c\leftarrow n}(t) = \psi^+_{c\leftarrow n}(t) + \gamma^+_{c\leftarrow n}(t) \tag{III.15}$$

$$f^-_c(t) = V_c\tau_c(t) - \sum_{n\in\mathcal{N}(c)} \left(\gamma^-_{c\leftarrow n}(t) + \psi^-_{c\leftarrow n}(t)\right) + E_c\psi_{c\downarrow}(t) \tag{III.16}$$

in which $\psi^+_{c\leftarrow n}(t)$ and $\gamma^+_{c\leftarrow n}(t)$ correspond respectively to the terms $\psi_{c\leftarrow n}(t)$ and $\gamma_{c\leftarrow n}(t)$ when they are positive and to 0 otherwise. Similarly, $\psi^-_{c\leftarrow n}(t)$ and $\gamma^-_{c\leftarrow n}(t)$ correspond to the terms $\psi_{c\leftarrow n}(t)$ and $\gamma_{c\leftarrow n}(t)$ when they are negative and to 0 otherwise.

This system of N^2 differential equations enables us to compute at each time point t, and for each cell c, the concentration $a_{c\leftarrow c'}(t)$ of auxin molecules in c that went at an anterior date through a given cell c', for any c'. This system of differential equations is solved using the finite difference method described above.

Contribution of a cell to the auxin concentration in particular zones Using the functions $a_{c\leftarrow c'}(t)$, it is possible to estimate the contribution of a particular cell to the overall auxin concentration of a group of cells (primordia or summit cells) in the meristem

at time t. For each cell c', its contribution to the auxin in a group G of cells, $\tau_{G \leftarrow c'}(t)$, is defined as:

$$\tau_{G \leftarrow c'}(t) = \frac{1}{\sum_{c \in G} q_c(t)} \sum_{c \in G} q_{c \leftarrow c'}(t) \tag{III.17}$$

where $q_{c \leftarrow c'}(t) = V_c a_{c \leftarrow c'}(t)$.

For a given group of cells G, e.g. the cells of a primordium, it is then possible to compute color maps showing a color intensity in each cell c' in the meristem proportional to the contribution $\tau_{G \leftarrow c'}(t)$ of this cell to the group. These maps give an impression of the main paths followed by auxin to reach particular zones in the meristem (see Figure 7 in main text).

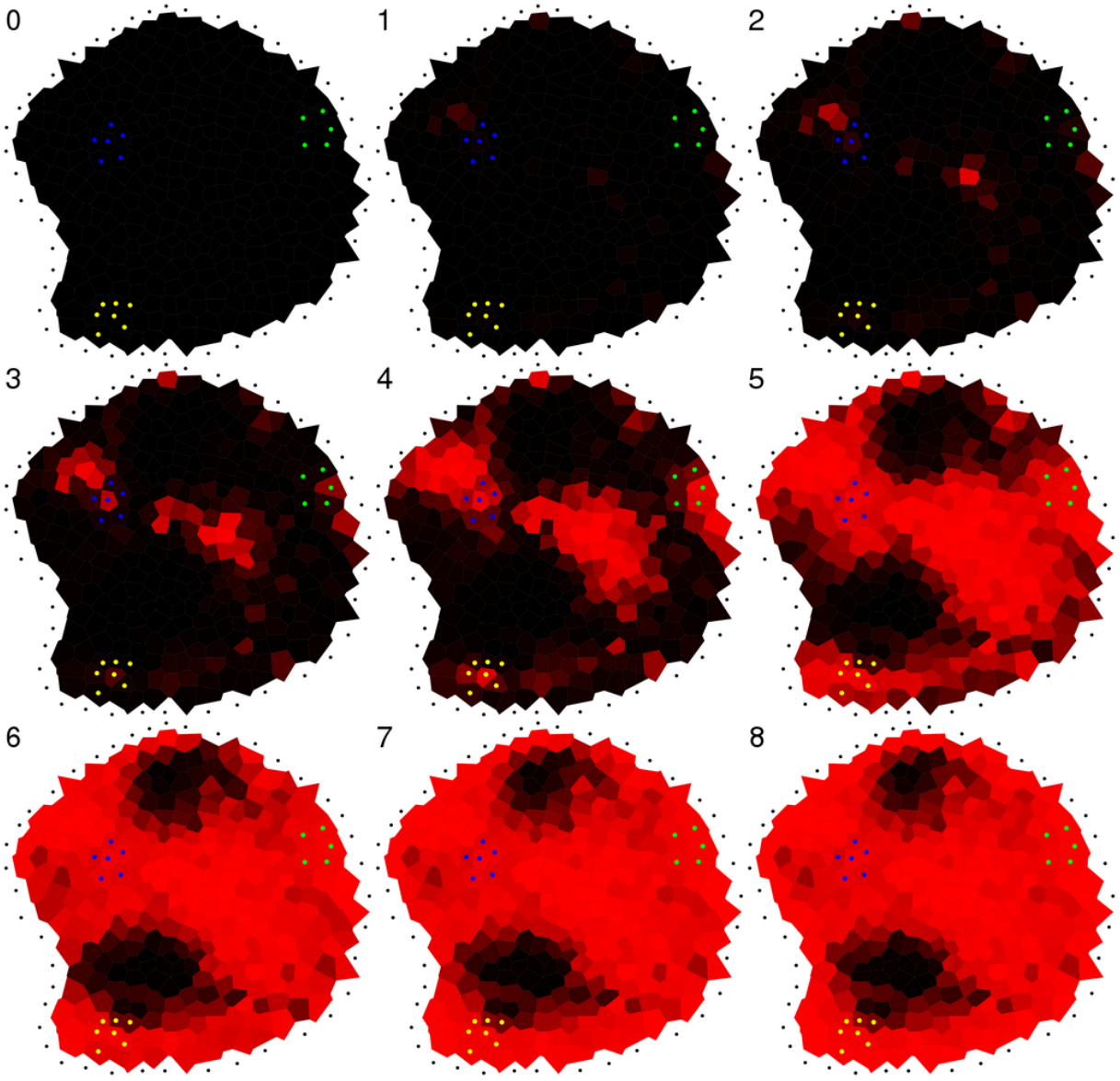


FIG. III.8: **Variation of auxin injection rate in a typical meristem.** These images show the different auxin patterns observed with the auxin injection rate varying following the values of table III.1.

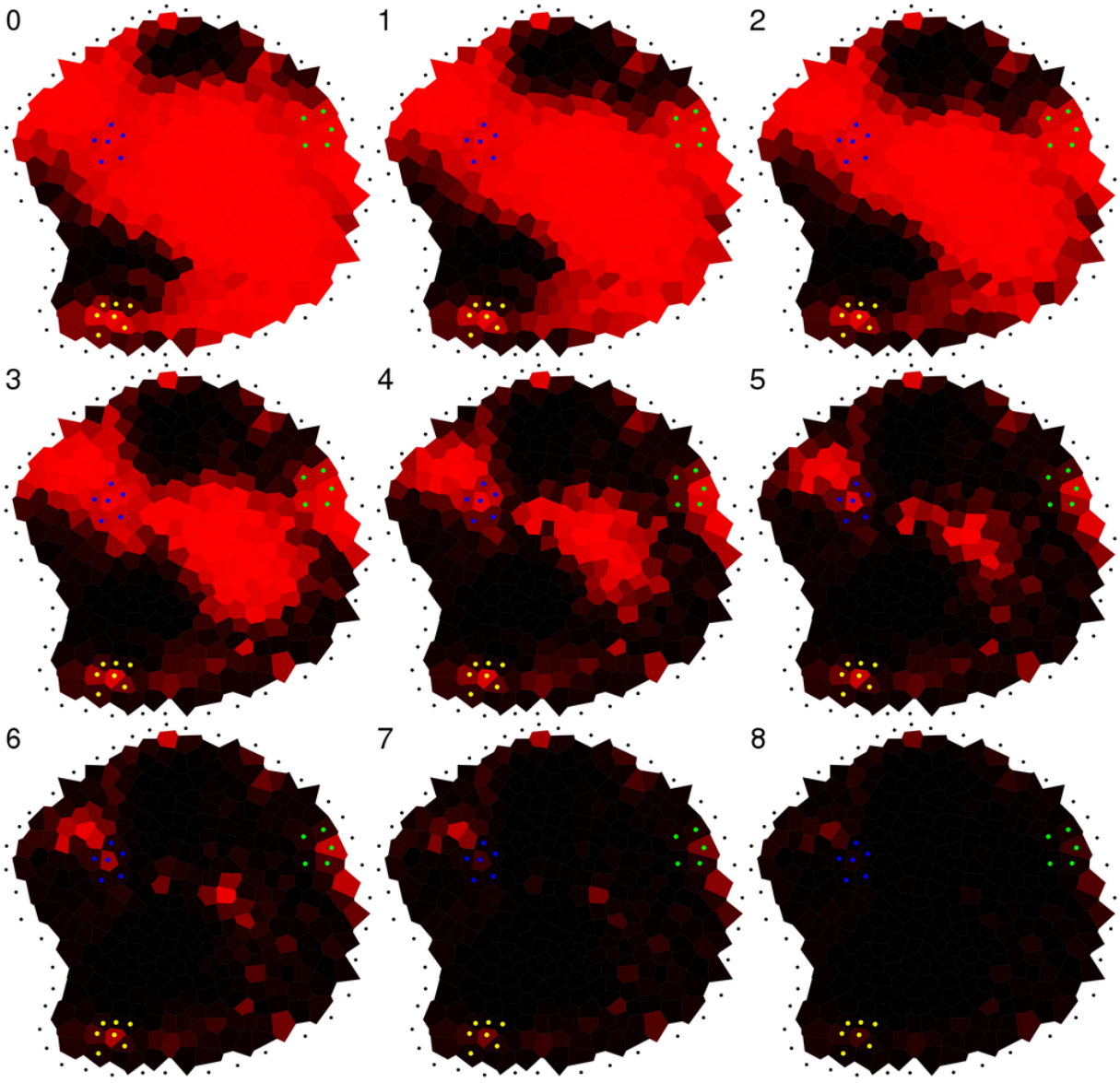


FIG. III.9: **Variation of the auxin degradation factor in a typical meristem.** These images show the different auxin patterns observed with the degradation factor varying following the values of table III.1.

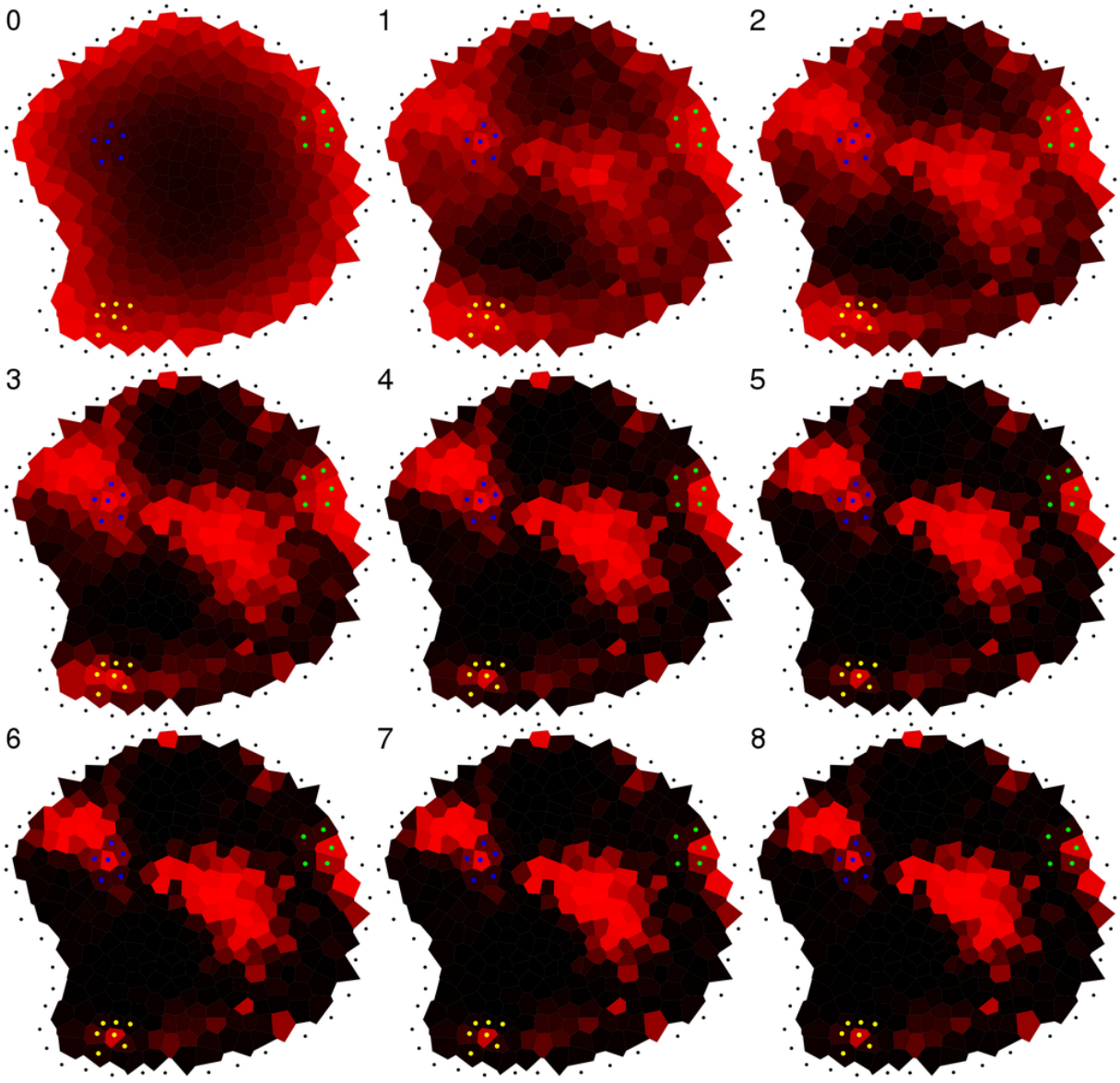


FIG. III.10: **Variation of auxin transport strength in a typical meristem.** These images show the different auxin patterns observed with the pumping strength varying following the values of table III.1.

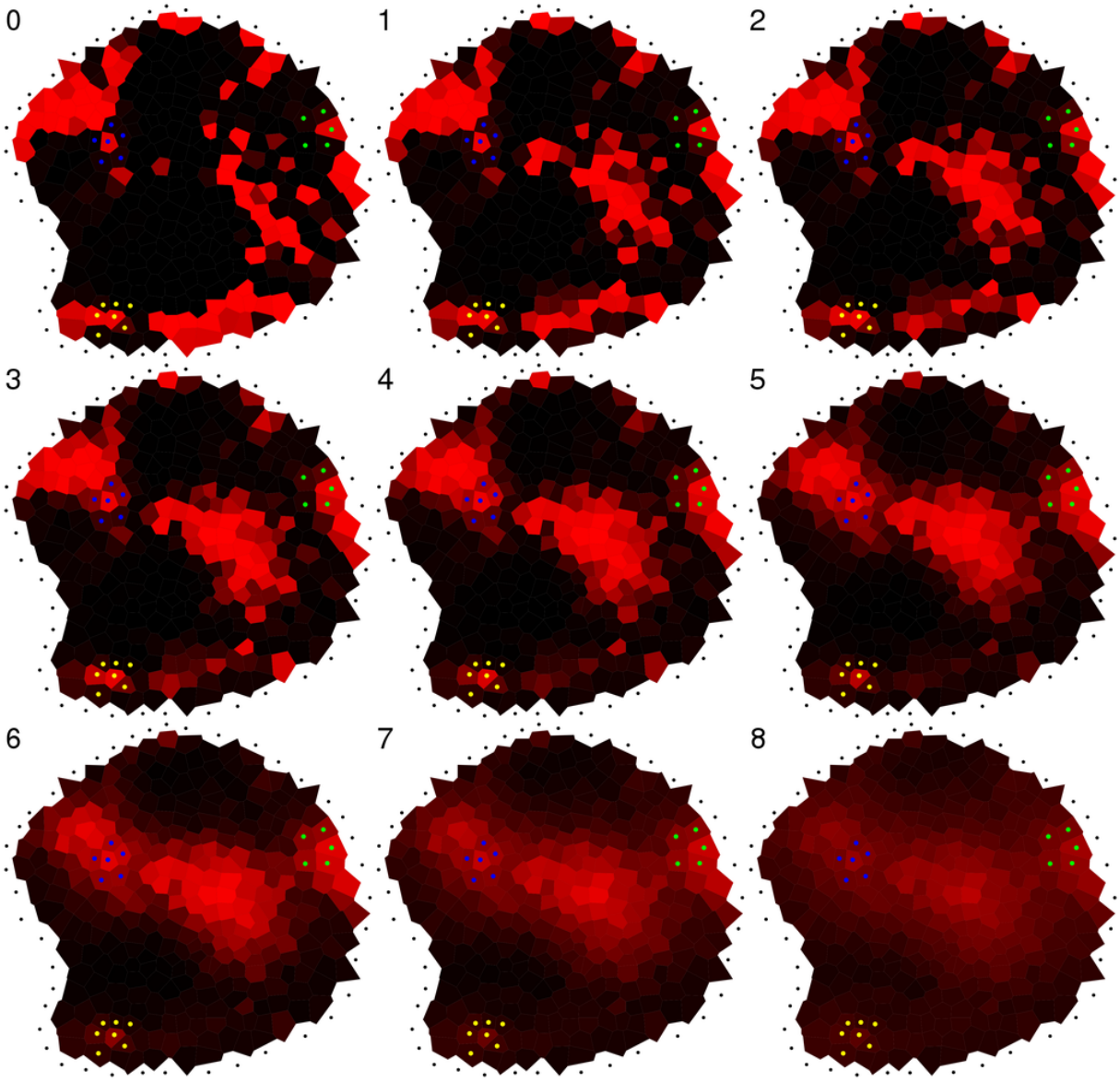


FIG. III.11: **Variation of diffusion factor in a typical meristem.** These images show the different auxin patterns observed with the diffusion factor varying following the values of table III.1.

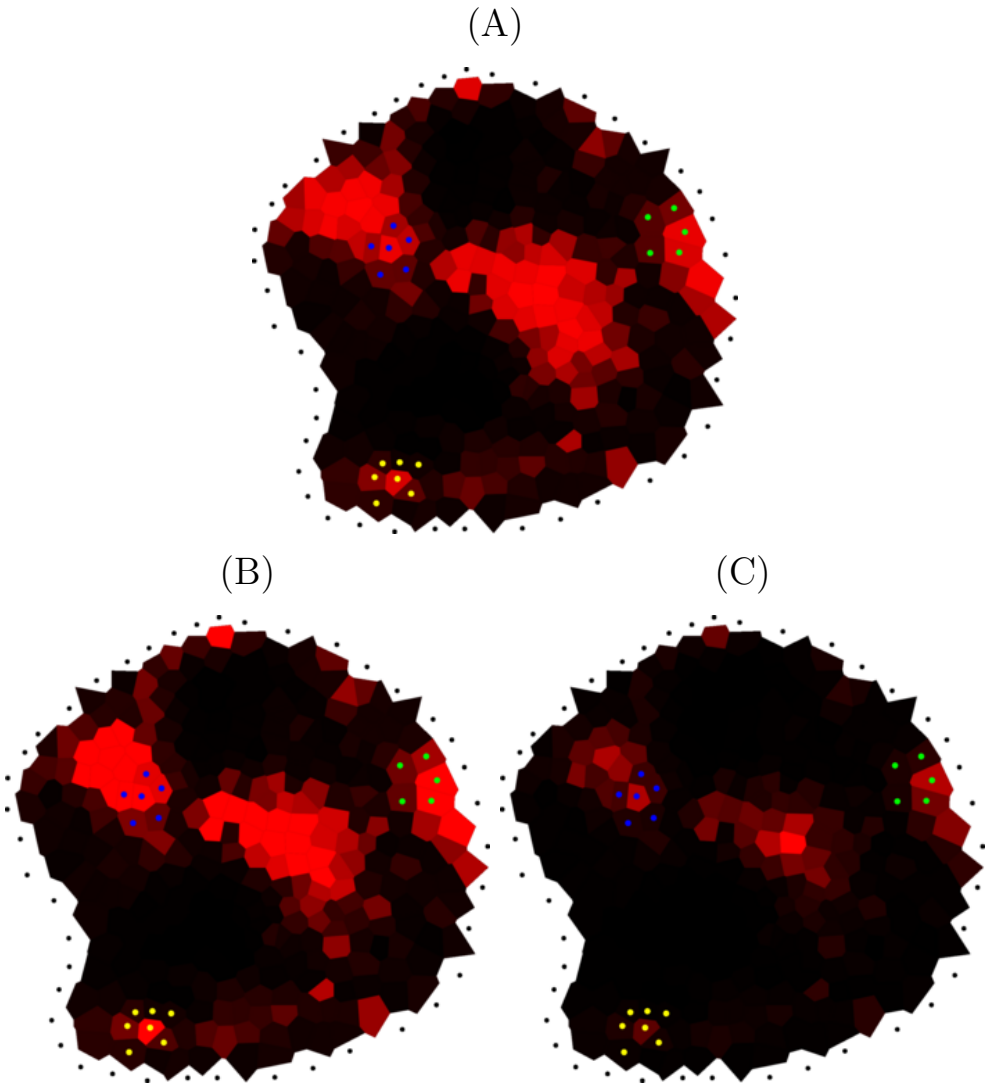


FIG. III.12: **Impact of saturation.**
(**A**) represents the result of the simulation with the reference values for parameters.
(**B**) Without saturation, the general auxin pattern does not vary significantly. Same color scale as in (A), note that, using this scale, a large amount of cells are above the maximum representable auxin concentration (ie. brightest red).
(**C**) Without saturation but here, the color scale was changed to fit the whole auxin concentrations range. This shows that auxin concentration at the meristem summit is higher than in primordia.

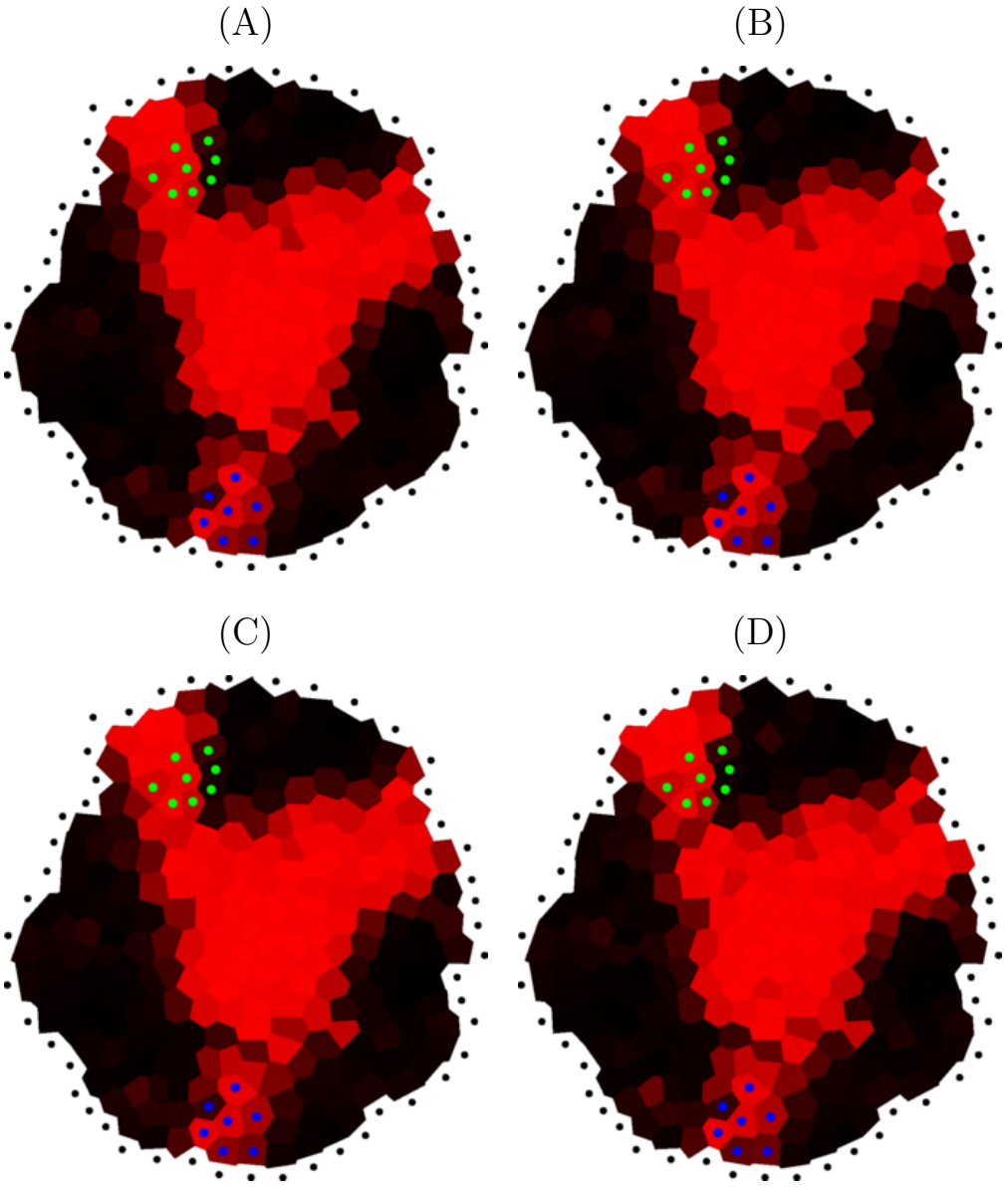


FIG. III.13: **Impact of the removal of less confident pumps.**
These images show the results of the simulations:
(A) with all the pumps;
(B) with only the pumps whose confidence level are 1, 2 or 3;
(C) with only the pumps whose confidence level are 1 or 2;
(D) with only the pumps with highest confidence level.

Chapitre IV

Modèle dynamique du méristème

Au chapitre III, nous avons développé un ensemble d'hypothèses permettant de simuler des flux d'auxine dans un méristème à un instant donné. L'étape suivante consiste à faire évoluer dans le temps le tissus méristématique et les concentrations d'auxine. Pour ce nouveau problème, les données biologiques sont moins nombreuses et nous devrons faire des hypothèses plus fortes que précédemment.

Nous proposons de voir dans ce chapitre tout d'abord les choix de modélisation qui ont été effectués, puis nous étudierons les résultats issus des simulations.

IV.1 Modélisation

Les récentes avancées dans les connaissances sur le fonctionnement du méristème apical caulinaire portent le plus souvent sur le comportement individuel des cellules et sur les échanges hormonaux entre les cellules. Aussi, pour simplifier la traduction de ces connaissances dans un modèle informatique, nous avons choisi de modéliser le tissu méristématique à l'échelle cellulaire.

Ce choix implique que nous devons modéliser les flux d'auxine sur une structure cellulaire en constante évolution. Le calcul des flux (et des concentrations) d'auxine dans le méristème correspond à la résolution d'un système dynamique classique, dans lequel l'état est défini par la concentration d'auxine dans chaque cellule et par l'intensité des flux d'auxine entre les cellules. Mais dans notre cas, il faut ajouter la structure cellulaire car son évolution n'est possible que si elle devient partie intégrante de l'état du système. Quand la structure fait partie de l'état, on parle de "système dynamique à structure dynamique" ($(DS)^2$; Giavitto *et al.*, 2002). Cette catégorie de modèle n'est étudiée que depuis relativement peu de temps. Il existe des outils spécialisés sur un type de structure particulier (par exemple les L-systèmes pour modéliser l'évolution des chaînes* et de certains arbres) et les recherches sur des outils mathématiques et informatiques plus génériques sont relativement récentes. La conception de ce $(DS)^2$ a donc été l'occasion de tester l'utilisation d'outils informatiques développés spécifiquement pour la résolution de ces modèles, mais cet aspect sera développé en section IV.2.

L'évolution du méristème est particulièrement complexe. D'une part, les concentrations d'auxine dépendent de la structure du méristème (et principalement du positionnement de la protéine PIN1). D'autre part, la structure du méristème est dépendante des concen-

trations d'auxine, en modifiant le positionnement de la protéine PIN1 (au moins par la création d'un primordium) ou en augmentant la croissance cellulaire. Ce modèle est donc un $(DS)^2$ où le développement de la structure et le fonctionnement interagissent.

Face à ce problème complexe nous avons distingué trois sous-problèmes relativement indépendants :

1. représenter le méristème
2. modéliser les interactions physiques entre les cellules
3. modéliser la physiologie des cellules et les échanges d'hormones

Nous commençons par le choix de la représentation du méristème car il conditionnera fortement les modèles utilisés par la suite. Nous avons cherché une représentation qui soit facile à manipuler et sur laquelle il est assez simple d'exprimer les connaissances biologiques dont on dispose. Nous avons ensuite décidé de scinder la modélisation de l'évolution du méristème en deux parties en faisant l'hypothèse que l'état physiologique du méristème est quasi-stationnaire par rapport à l'état mécanique. Autrement dit, nous supposons que les interactions physiques sont infiniment plus lentes que les processus physiologiques. En particulier, on peut observer l'émergence de trois primordia par jour en moyenne sur un apex d'*Arabidopsis*, alors que l'auxine diffuse à travers l'ensemble du méristème en quelques minutes (Reinhardt *et al.*, 2000).

Nous allons maintenant exposer les choix et leurs justifications pour chacun des sous-problèmes identifiés.

IV.1.1 Représentation du méristème

Nous distinguons deux catégories dans les représentations à l'échelle tissulaire : les représentations discrètes et les représentations continues. Le choix d'une ou l'autre représentation dépend d'une part du type de connaissance dont on dispose, et de la vue qu'on adopte par rapport aux tissus végétaux. Une première approche des tissus végétaux consiste à les considérer comme un volume continu dont les parois assurent la rigidité. Cette approche est cohérente avec l'usage d'un modèle continu de tissu cellulaire pour modéliser le tissu. La seconde approche consiste à considérer les tissus comme un ensemble cohérent de cellules autonomes et communicantes. Cette second approche est plus cohérente avec l'usage d'un modèle discret (pour discussion : Kaplan et Hagemann, 1991).

Les informations dont nous disposons sont exprimées cellules par cellules (i.e. positionnement de la protéine PIN1, activité des gènes...), avec une approche du tissu comme un ensemble de cellules. Aussi, pour faciliter l'utilisation de ces informations dans notre modèle, nous avons opté pour une représentation discrète à l'échelle cellulaire.

Le problème de la représentation d'un tissu animal ou végétal à l'échelle cellulaire a déjà été largement abordé dans le passé. Les supports les plus utilisés incluent (pour revue : Lantin, 1996) :

– les objets de taille nulle (points, symboles ...) (Lindenmayer, 1968)
  Dans ces représentations, les objets n'ont pas d'encombrement et ont une relation topologique simple (séquence, ensemble...). En cas de placement géométrique il doit être imposé de l'extérieur (sur une grille, répulsion...)

- les graphes topologiques (Matela et Fletterick, 1979)
  Dans ces représentations, seule la relation topologique entre les cellules est explicitée. La forme, la taille, la position ne sont pas prises en compte.
- les polygones ou polyèdres (Lück et Lück, 1982)
  Ce sont des représentation duales* des précédentes. Seule la géométrie des parois cellulaires est explicitée. La topologie est, si nécessaire, déduite à partir de la géométrie par la notion de contact entre polygones.
- les diagrammes de Voronoï (Honda, 1978, 1983, 1994)
  Dans cette représentation, ni la géométrie, ni la topologie ne sont explicitées. Seul un ensemble de points dans l'espace est explicité, ensemble dont on déduit la topologie (le graphe de Delaunay) et la géométrie (les domaines de Voronoï).
- les disques ou sphères (Bodenstein, 1986; Jönsson *et al.*, 2005)
  Par rapport à l'usage de polygones, les disques ou sphères présentent l'avantage de ne pas nécessiter la maintenance d'une géométrie complexe, mais permettent tout de même de prendre en compte l'encombrement des cellules.

Notre objectif premier pour le choix d'une représentation est de permettre d'exprimer son évolution le plus simplement possible, tout en maintenant une géométrie et une topologie cohérente. Aussi, nous avons choisi d'utiliser les diagrammes de Voronoï. Par ailleurs, nous prenons l'hypothèse que les processus régulant le positionnement des organes se déroulent dans la couche superficielle de cellules (voir section I.1.2), nous utiliserons donc des diagrammes de Voronoï en dimension deux (appelés parfois domaines de Dirichlet) et nous ne représenterons pas explicitement l'intérieur du méristème.

IV.1.1.1 Diagramme de Voronoï et graphe de Delaunay

Le concept de diagrammes de Voronoï est apparu dès 1644 dans *Le Monde de Mr. Descartes, ou Le Traité de la Lumière*, où René Descartes les utilise pour l'étude des zones d'influence gravitationnelle des étoiles (source : Hoff III *et al.*, 1999). La première formalisation est réalisée par Lejeune-Dirichlet (1850). Le mathématicien allemand a étudié les cas 2D et 3D, auxquels ont donne parfois le nom de tesselation de Dirichlet. L'appellation "diagramme de Voronoï" vient du mathématicien russe Voronoï (1908) qui a étendu le concept aux dimensions supérieures. Depuis, le concept a été redécouvert plusieurs fois et les domaines d'application des diagrammes de Voronoï sont très nombreux (pour revue : Aurenhammer, 1991).

Définition IV.1 (Diagramme de Voronoï) *Formellement, dans un espace euclidien, le diagramme de Voronoï d'un ensemble de points $\mathcal{S}$ représente la classe d'équivalence de la relation "avoir même plus proches voisins dans $\mathcal{S}$". Soit un ensemble $\mathcal{S} = \{S_1, S_2, \ldots, S_n\}$ de n points de $\mathbb{E}^d$, qu'on appellera* sites. *On associe à chaque site S_i la région $V(S_i)$ de $\mathbb{E}^d$ constituée des points plus proches de S_i que des autres sites:*

$$V(S_i) = \left\{ X \in \mathbb{E}^d / d(X, S_i) \leq d(X, S_j), \forall j \neq i \right\} \tag{IV.1}$$

Le diagramme de Voronoï est défini comme l'ensemble des régions $V(S_i)$.

Le graphe de Delaunay (inventé par le mathématicien russe Delaunay) peut être défini comme le dual* du diagramme de Voronoï.

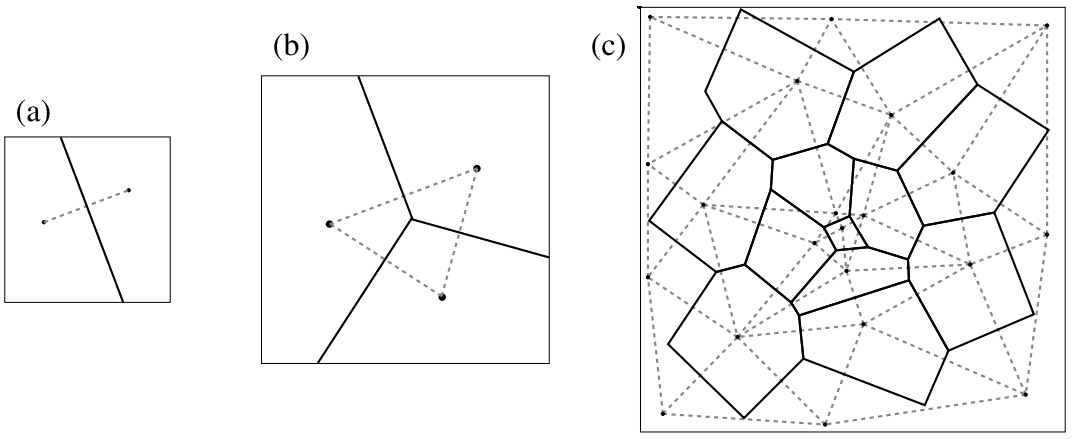


FIG. IV.1: **Diagramme de Voronoï et graphe de Delaunay.** Le graphe de Delaunay (en pointillés gris) relie les sites (points noirs) donc les régions de Voronoï (traits pleins) sont adjacentes.
(a) Lorsqu'il n'y a que deux sites, la région de Voronoï de chaque site est le demi-espace délimité par la médiatrice du segment reliant les sites.
(b) Avec trois sites, ce sont toujours les médiatrices qui délimitent les domaines de Voronoï de chaque site.
(c) De manière plus générale, les domaines de Voronoï sont délimités par les médiatrices des segments qui forment des arcs dans le graphe de Delaunay. Dans ce dessin, seules les régions bornées sont représentées.

Définition IV.2 (Graphe de Delaunay) *Le graphe de Delaunay est défini par la relation d'adjacence entre les régions du diagramme de Voronoï. C'est donc un graphe non dirigé. Les nœuds du graphe sont les points de $\mathcal{S}$, et l'application $\mathcal{N}$ qui associe à chaque nœud du graphe l'ensemble de ses voisins dans le graphe est défini par:*

$$\mathcal{N}(S_i) = \{S_j / V(S_i) \cap V(S_j) \neq \emptyset, \forall j \neq i\} \tag{IV.2}$$

Définition IV.3 (L_2-généralité) *En dimension d, un ensemble de points est dit en position L_2-générale s'il n'existe pas de sphère passant par $2 + d$ d'entre eux.*

Par la suite, nous ne considérerons que le cas en deux dimensions. Pour les algorithmes et définitions dans le cas général, voir Boissonnat et Yvinec (1995), cinquième partie.

Construction du graphe de Delaunay et du diagramme de Voronoï. Un moyen simple de construire le graphe de Delaunay d'un ensemble de sites est de partir sur le théorème suivant (voir figure IV.2) :

Théorème IV.1 *Si les sites sont en position L_2-générale, alors trois sites S_i, S_j, S_k sont adjacents dans le graphe de Delaunay si et seulement si le disque circonscrit au triangle $S_iS_jS_k$ ne contient aucun autre site.*

Une première méthode consiste donc à relier tous les triplés de sites qui sont sommets d'un triangle tels que le cercle circonscrit au triangle ne contienne aucun autre site. Cette

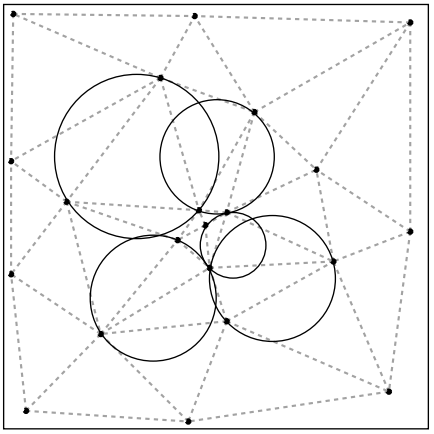


FIG. IV.2: **Cercle circonscrit à un triangle de sommets adjacents.**Le cercle circonscrit au triangle formé par trois sites adjacents ne contient aucun autre site (théorème IV.1).

méthode a l'avantage d'être simple, mais elle est inefficace. Aussi, nous utilisons un autre algorithme décrit avec précision dans Boissonnat et Yvinec (1995), chapitre 17 et implémenté dans la bibliothèque QHull (Barber *et al.*, 1996). La complexité de cet algorithme est en $O(n\log(n) + n^{d/2})$, où n est le nombre de sites et d la dimension de l'espace, ce qui est la complexité optimale pour un algorithme calculant le graphe de Delaunay.

À partir du graphe de Delaunay, le diagramme de Voronoï s'obtient facilement en utilisant les théorèmes suivants.

Théorème IV.2 *Si les sites sont en position L_2-générale, les sommets de la région de Voronoï d'un site S_i sont les barycentres des triangles $S_iS_jS_k$ où S_i, S_j et S_k sont voisins deux à deux dans le graphe de Delaunay.*

Théorème IV.3 *Les régions du diagramme de Voronoï sont des convexes, éventuellement non-bornés.*

Ces deux théorèmes permettent de construire toute région bornée du diagramme de Voronoï à partir du graphe de Delaunay. Nous n'aborderons pas le problème des régions non-bornées car nous n'aurons pas à les reconstruire.

IV.1.1.2 Représentation du tissu méristématique

Ainsi, le couple diagramme de Voronoï, graphe de Delaunay nous permet, à partir simplement d'un ensemble de points, de maintenir une géométrie et une topologie cohérentes.

Notre objectif est de représenter chaque cellule c par un site S_c, la région de Voronoï $V(S_c)$ correspondant à la forme de la cellule, les sites étant répartis dans un disque de rayon r_M représentant la surface du méristème. Par ailleurs, on note d_m la distance minimale entre deux sites.

Les diagrammes de Voronoï possèdent une caractéristique qu'il est nécessaire de prendre en compte si l'objectif est de représenter des cellules,. Une partie des sites représentant

les cellules au bord du tissus se verront associées une région infinie. Pour éviter de cela, nous ajoutons aux sites représentant les cellules, d'autres sites qui les encadrent de façon à ce que toutes les régions infinies leur soient associées : nous appellerons ces nouveaux sites *ancres*. Le placement des ancres doit, idéalement, induire des tailles de cellules raisonnables pour les cellules adjacentes (i.e. leur taille doit être comparable à la taille moyenne des autres cellules). Dans la suite, les sites représentant les cellules seront appelés *centroïdes* des cellules.

Nous avons choisi de créer un méristème virtuel plan, entièrement contenu dans un disque de rayon r_M avec une cellule placée en son centre, appelée cellule centrale du méristème. Les ancres sont équiréparties sur un cercle de rayon $r_M + \delta_a$ espacées d'au plus $2\delta_a$, où δ_a doit être de l'ordre de la distance moyenne entre deux cellules.

IV.1.2 Modèle physique

Dans l'optique d'une modélisation à l'échelle cellulaire, nous avons choisi de modéliser la dynamique du tissu en considérant la croissance et la division des cellules individuellement, le déplacement apparent des cellules et la croissance du tissu étant le résultat des interactions mécaniques locales entre cellules.

Plus précisément, nous modélisons ces interactions mécaniques par un système visco-élastique (e.g. Boas, 1983), les ressorts reliant les centroïdes des cellules adjacentes. Nous choisissons de fixer la raideur de tous les ressorts à une même valeur. Dans ce modèle, la croissance est due à l'augmentation de la longueur à vide des ressorts.

Par rapport à un système visco-élastique classique, nous imposons deux contraintes. La première est que la taille de chaque cellule fait partie de son état et est représentée par une grandeur unique, variable dans le temps. La seconde est due au choix du diagramme de Voronoï : le voisinage d'une cellule étant susceptible de changer, le système doit être défini indépendamment du voisinage exact de chaque cellule.

Une solution consiste à relier les centroïdes de cellules voisines par un ressort composé de deux demi-ressorts. La taille des demi-ressorts l_c associés à une cellule c donnée est constante et représente son paramètre de taille. Les centroïdes de deux cellules c_1 et c_2 adjacentes sont alors reliés par un ressort r_{c_1,c_2} donc la longueur à vide totale l_{c_1,c_2} est la somme des deux demi-ressorts associés à chacune des cellules :

$$l_{c_1,c_2} = l_{c_1} + l_{c_2} \tag{IV.3}$$

La force $\vec{F_c}$ exercée par les ressorts sur une cellule c est alors donnée par :

$$\vec{F_c} = -\xi \vec{V_c} + \sum_{n \in \mathcal{N}(c)} K(d(c,n) - l_{c,n}) \vec{u}_{c,n} \tag{IV.4}$$

où ξ est la viscosité du frottement fluide, $\vec{V_c}$ est la vitesse du centroïde de la cellule c, K est la raideur des ressort, $d(c,n)$ est la distance entre la cellule c et la cellule n, $\vec{u}_{c,n}$ est le vecteur unité colinéaire au vecteur allant du centroïde de la cellule c à celui de la cellule n et $\mathcal{N}(c)$ est l'ensemble des cellules voisines de c.

Dans ce système, la croissance d'une cellule est représentée par un allongement du demi-ressort qui lui est associé. La taille d'une cellule augmente si son centroïde s'éloigne

des centroïdes voisins. Il faut noter que la notion d'éloignement étant symétrique, la croissance d'une cellule provoque en général une croissance chez les cellules voisines. Nous avons choisi d'augmenter les longueurs de chaque ressort d'une quantité constante à chaque pas de temps de façon à assurer une croissance uniforme sur l'ensemble du méristème.

Le système de résolution du système d'équations différentielles est identique à celui employé pour le modèle de flux dans les méristème réels, à savoir un schéma explicite d'intégration de Euler à pas de temps constant (Press *et al.*, 1992).

En plus de la croissance cellulaire, il est nécessaire de modéliser la division cellulaire. Lors de la création du méristème, chaque cellule se voit affecter une taille et une direction (radiale ou orthoradiale) de division. Quand la surface d'une cellule dépasse sa taille critique, elle se divise selon la direction qui lui a été affectée et crée deux cellules filles qui se diviseront selon l'autre direction. Chaque cellule fille se voit attribuer une taille critique qui est choisie aléatoirement et indépendamment l'une de l'autre.

Enfin, pour limiter le temps de calcul, le système ne conserve que les cellules qui restent dans un disque de taille défini centré sur le méristème. Afin de limiter l'effet de la suppression des cellules et notamment l'apparition d'une rotation globale du méristème, les sites des cellules extérieures du méristème observé ne peuvent se déplacer que radialement. Une cellule est considérée comme extérieure si elle est voisine d'une ancre dans le graphe de Delaunay.

IV.1.3 Modèle physiologique

Notre modèle physiologique comporte trois étapes : (i) la répartition et le positionnement de la protéine PIN1, (ii) le transport de l'auxine et (iii) la création des primordia.

Ces trois étapes sont gérées indépendamment les unes des autres car elles sont supposées se dérouler à des échelles de temps différentes. Premièrement, le repositionnement de PIN1 est supposé se faire instantanément (i.e. à l'échelle de temps la plus petite). Ensuite, le transport d'auxine est supposé être suffisamment rapide pour atteindre son état stationnaire avant qu'un primordium puisse être créé. Enfin la création des primordia ne se fait que lorsque tout le reste est stable.

La résolution du modèle physiologique se fait en itérant sur ces trois étapes en prenant en compte les échelles de temps différentes jusqu'à ce qu'il n'y ait plus aucun changement. Alors seulement, une nouvelle étape mécanique est initiée pour un nouveau cycle complet de calcul.

IV.1.3.1 Placement de la protéine PIN1

Du fait des techniques d'observations à notre disposition, nous n'avons pas d'information sur la dynamique du placement de la protéine PIN1. Aussi, nous avons décidé d'utiliser une conséquence indirecte de l'organisation de la protéine PIN1.

Au chapitre III, nous avons présenté une méthode de calcul de la contribution de chaque cellule à l'auxine présente dans les primordia et le centre. Ceci nous a conduit à observer le résultat suivant : généralement, une cellule envoie la plupart de son auxine vers une destination unique (i.e. primordium ou centre). De plus, les cellules envoyant leur auxine vers un même élément sont regroupées et forment un ensemble connexe. Ainsi, il

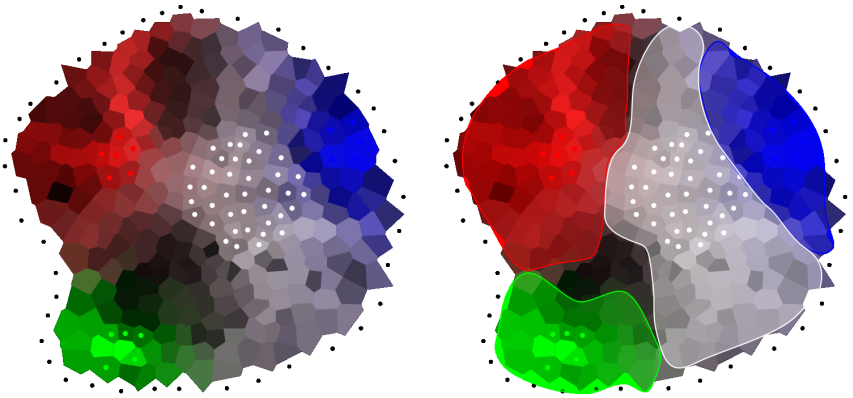


FIG. IV.3: **Zones d'influences dans un méristème réel.** La figure de gauche représente la participation de chaque cellule à la concentration d'auxine du centre (en blanc) ou d'un des primordia (en bleu, rouge ou vert). Les primordia sont indiqués par un point de couleur dans les cellules en faisant partie, disque de la même couleur que celle utilisée pour la participation des cellules à ce primordium. Les cellules du centre sont indiquées par un disque blanc en leur centre.
La figure de droite ajoute les zones d'influence de chaque élément en une zone semi-transparente de la couleur de l'élément.

est possible de définir la zone d'influence de chaque élément comme l'ensemble des cellules qui envoient leur auxine principalement vers cet élément (voir figure IV.3).

Notre objectif est de reproduire les zones d'influences observées dans les cartes de suivi d'auxine. Une solution consiste à placer la protéine PIN1 selon une règle de magnétisation des différents "attracteurs" (i.e. le centre et les primordia).

Dans cette solution, chaque primordium exerce un potentiel d'attraction sur une cellule donnée selon la loi :

$$\lambda_{c,p} = \Lambda_p - d(c,p)^2 \tag{IV.5}$$

où $\lambda_{c,p}$ est la force d'attraction du primordium p sur la cellule c, Λ_p est la force d'attraction de p et $d(c,p)$ est la distance entre c et p.

Quant au centre, son potentiel d'attraction sur une cellule est donné par la loi :

$$\lambda_{c,c_0} = \max(0, \Lambda_{c_0} - d(c,c_0)) \tag{IV.6}$$

où c_0 est la cellule qui se trouve au centre du méristème (voir section IV.1.1.2) et Λ_{c_0} est la force d'attraction du centre du méristème.

Une cellule est alors attirée uniquement par l'attracteur dont le potentiel d'attraction sur elle est le plus fort. La protéine PIN1 est ensuite placée en direction de la cellule qui est le plus proche de la direction de l'attracteur. Ainsi, si on note $p_1, p_2, \ldots, p_n$ les primordia, alors la protéine PIN1 dans la cellule c est placée sur la paroi entre les cellules c et c', où c' est déterminée par (voir figure IV.4):

$$p = \arg\max_{p \in \mathcal{A}} \lambda_{c,p} \quad \text{où } \mathcal{A} = \{c_0, p_1, p_2, \ldots, p_n\} \tag{IV.7}$$

$$c' = \arg\min_{n \in \mathcal{N}(c)} \widehat{pcn} \tag{IV.8}$$

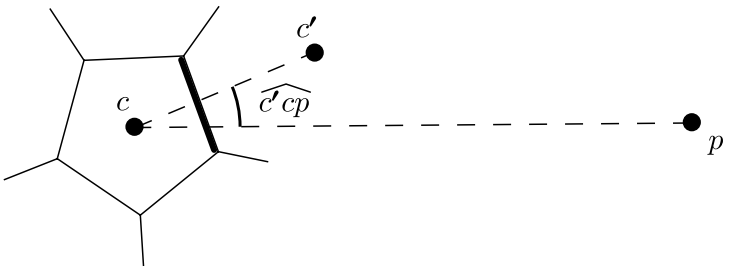


FIG. IV.4: **Orientation de PIN1 dans une cellule.** La protéine PIN1 est positionnée dans la cellule c vers la cellule c' qui est le plus dans la direction de l'attracteur p le plus fort de c. PIN1 est indiquée par un trait noir épais sur la paroi qui la porte.

IV.1.3.2 Transport de l'auxine

Le transport de l'auxine se fait exactement comme décrit dans le chapitre III. Pour mémoire, les principales hypothèses sont :

(i) le transport d'auxine est composé d'une diffusion à travers les parois et d'un transport actif par l'intermédiaire de la protéine PIN1.

(ii) hors des primordia, le transport d'auxine n'est significatif que dans la couche de cellules L1.

(iii) dans les primordia, l'auxine est évacuée par la protéine PIN1 vers les couches cellulaires internes du méristème.

(iv) l'auxine est produite par des parties sous-jacentes au méristème et arrive par la couche L1 dans le méristème.

Comme nous l'avons dit plus haut, nous supposons que le positionnement de la protéine PIN1 est un processus bien plus rapide que le transport de l'auxine. Il nous faut donc nous assurer que la protéine PIN1 est toujours positionnée correctement entre deux mise-à-jour des concentrations d'auxine.

Toutefois, le placement de la protéine étant indépendant de la concentration d'auxine dans ce modèle, nous n'avons pas à recalculer sa position tant que l'auxine n'a pas induit d'autre modification (en l'occurrence l'apparition d'un primordium).

IV.1.3.3 Initiation des primordia

Les primordia sont initiés uniquement dans un anneau autour de la zone centrale du méristème appelé zone de compétence. Pour qu'un primordium se forme, il faut que la concentration d'auxine dans une cellule et ses voisines dépasse un certain seuil.

La zone de compétence est définie comme l'ensemble des cellules qui sont hors d'un disque de rayon r centré sur le méristème mais qui ont au moins une voisine dans ce disque :

$$ZC = \{c \in \mathcal{M} \,|(d(c, c_0) > r) \wedge (\exists n \in \mathcal{N}(c)/d(n, c_0) < r)\} \tag{IV.9}$$

où $\mathcal{M}$ est l'ensemble des cellules du méristème virtuel, c_0 est la cellule centrale du méristème et $\mathcal{N}(c)$ est l'ensemble des voisines de la cellule c.

Une cellule c peut devenir primordium si sa concentration a_c en auxine et celle de ses voisines sont supérieures à un seuil σ_{Pr}:

$$(a_c > \sigma_{Pr}) \wedge (\forall n \in \mathcal{N}(c) : a_n > \sigma_{Pr}) \tag{IV.10}$$

On détermine alors l'ensemble des cellules susceptibles de devenir primordium :

$$\widetilde{\mathcal{P}r} = \{c \in ZC | \, (a_c > \sigma_{Pr}) \wedge (\forall n \in \mathcal{N}(c) : a_n > \sigma_{Pr})\} \tag{IV.11}$$

Parmi ces cellules, celle qui concentre le plus d'auxine devient primordium :

$$\mathcal{P}r(c) \Leftarrow \arg\max_{c' \in \widetilde{\mathcal{P}r}}(a_{c'}) \tag{IV.12}$$

où $\mathcal{P}r(c)$ est vraie si la cellule c est le centre d'un primordium.

IV.2 Paradigme de programmation pour la morphogénèse

Comme nous l'avons vu au début de ce chapitre, un tissu cellulaire évoluant dans le temps fait parti des $(DS)^2$. Or, les outils mathématiques et informatiques adaptés aux $(DS)^2$ font, actuellement, l'objet de recherches. Parmi les outils en développement, nous avons voulu évaluer l'apport d'un langage dédié.

Il existe peu de langages dédiés à l'expression des $(DS)^2$. Parmi les plus connus, les L-systems permettent d'exprimer l'évolution de structures telles que les séquences ou les arbres. Le modèle que nous construisons repose sur la notion de graphes de Delaunay : le langage que nous utilisons doit pouvoir permettre d'exprimer l'évolution d'un tel graphe. À notre connaissance, très peu de langages le permettent. MGS (Giavitto et Michel, 2001a) et GroIMP (Kniemeyer, 2004) en sont deux exemples. Les deux langages ont en commun de reprendre l'idée générale des L-systèmes à des structures plus générales. Ils proposent de décrire l'évolution d'un système complexe par l'évolution de ses sous-parties, ce qui impose d'une part de découper une structure en sous-parties et d'autre part d'exprimer les règles locales d'évolution. Alors que GroIMP s'est orienté vers la réécriture de graphes orientés et propose de ramener la plupart des structures de données classiques vers la structure de graphe, MGS a pris le parti de différencier les différentes structures de données et de proposer une manière unique d'exprimer l'évolution de structures différentes. Pour notre projet, nous avons décidé d'évaluer l'utilisation de MGS, notamment car l'implémentation du langage est plus mûre et qu'elle permet la manipulation de graphes de Delaunay.

IV.2.1 Présentation de MGS.

MGS est un langage qui propose d'unifier plusieurs paradigmes de programmation comme la transformation de multi-ensembles*, les L-systèmes ou les automates cellulaires. L'objectif de MGS est de permettre l'expression de l'évolution de structures complexes comme l'évolution des parties de ces structures.

C'est un langage fonctionnel*, enrichi avec de nouveaux types de valeurs structurées, les *collections topologiques*, et par une syntaxe pour définir des fonctions sur ces valeurs de manière déclarative, les *transformations*.

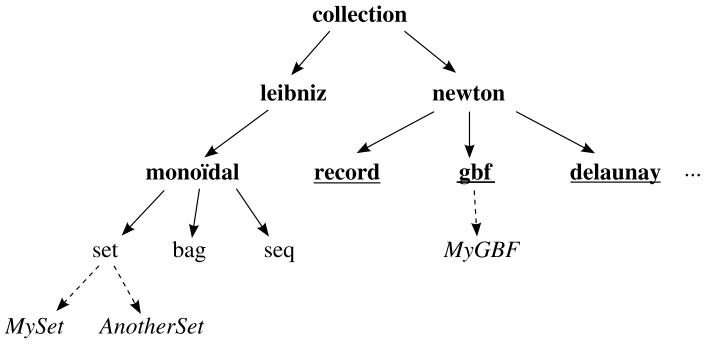


FIG. IV.5: **Hiérarchie des collections topologiques de MGS.** MGS propose une hiérarchie de collections topologiques. Cette collection comporte notamment des types abstraits (en gras), des types paramétriques (en souligné) et des sous-types définis par l'utilisateur (en italique).
Adapté de : (Giavitto et Michel, 2001a).

Les collections topologiques. Une collection correspond à un ensemble de *places* ou *positions* labellées par des valeurs. L'organisation d'une collection peut se décrire par sa topologie en définissant la relation de voisinage entre les positions. Une collection peut correspondre à une structure de donnée classique, telle qu'une liste ou un ensemble, ou à une structure plus complexe, telle qu'un graphe ou une grille.

MGS propose un ensemble de collections topologiques organisées en une hiérarchie (voir figure IV.5).

Parmi les collections proposées, les collections abstraites (i.e. dont il n'existe pas d'instance directe) permettent de regrouper les collections ayant des caractéristiques communes. Ainsi, un aspect important de la hiérarchie est que toutes les collections se répartissent entre les collections abstraites *newtoniennes* et *leibniziennes* (qui sont ainsi pratiquement au plus haut dans la hiérarchie des collections), du nom des deux philosophes Newton et Leibniz qui avaient chacun leur théorie sur la nature de l'espace (Giavitto et Michel, 2002). Pour Newton, l'espace pré-existe et joue le rôle d'une scène sur laquelle les objets sont placés. Ainsi, il est toujours possible de parle de position et de place dans un espace vide. Pour Leibniz, l'espace n'existe que par les objets qui le composent : c'est la relation positionnelle entre les objets présents. Dans cette vue, le concept de place ou de position dans un espace vide est inconcevable.

Les collections paramétriques, qui sont nécessairement abstraites, permettent de définir des types partiellement spécifiés que l'utilisateur pourra préciser pour créer un type concret. Par exemple, pour définir un graphe de Delaunay, il est nécessaire de définir le nombre de dimensions de l'espace dans lequel les positions sont exprimées et la fonction calculant la position d'un élément cet espace.

Enfin, MGS permet à l'utilisateur de définir de nouveaux sous-types qui auront exactement la même topologie que leur type parent. Dans cette hiérarchie, chaque instance d'un type concret et aussi instance de tous les parents de ce type.

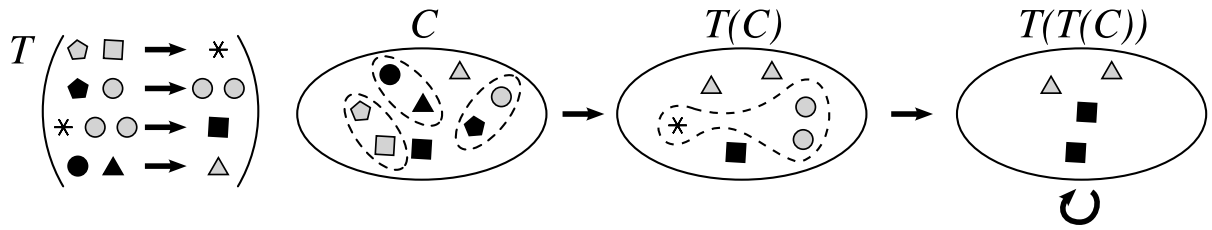


FIG. IV.6: **Exemple d'une transformation MGS.** La transformation T est appliquée à l'ensemble C plusieurs fois. À chaque étape, les ensembles disjoints d'éléments qui forment la partie gauche d'une des règles de T sont remplacés simultanément par les éléments de la partie droite de la règle.
Adapté de : Giavitto et Michel (2001a)

Les transformations. Les transformations sont des fonctions définies de manière déclarative, qui permettent d'exprimer l'évolution d'une collection topologique par la réécriture de chemins dans cette collection (voir figure IV.6). Un chemin est alors définit par une séquence de positions dans la collection telles que deux positions consécutives du chemin sont voisines dans la collection. Une transformation peut se définir par l'application en parallèle de plusieurs règles locales, chaque règle définissant la transformation d'un chemin dans la collection.

Une règle locale est composée d'un sélecteur, d'une fonction et de contraintes. Le sélecteur permet de définir un chemin dans la collection topologique sur lequel une propriété est vérifiée. La fonction admet comme argument les éléments nommés du chemin sélectionné et renvoie une nouvelle séquence qui remplacera le chemin sélectionné. Enfin, les contraintes permettent de préciser le comportement de la règle en cas de conflit (priorité, stratégie de désambiguïsation...).

Exemple d'utilisation. Nous allons présenter un programme simplifié permettant de modéliser la croissance d'un tissus composé d'une cellule apicale unique, de quelques cellules sub-apicales et de cellules végétatives. Le modèle mécanique est semblable à celui décrit en section IV.1.2, à ceci prêt qu'il n'y a pas d'ancres (car on se considère pas la taille des cellules) et que les cellules évoluent dans un espace à 3 dimensions.

Dans ce tissu, la cellule apicale se divise régulièrement pour produire deux cellules sub-apicales et une nouvelle cellule apicale. Chaque cellule sub-apicales produisent à leur tour trois cellules végétatives, et enfin les cellules végétatives croissent sans se diviser jusqu'à atteindre une taille maximale (voir figure IV.7).

Le programme ci-dessous se décompose en trois parties. La première (lignes 1 à 20) s'occupe de calculer l'état d'équilibre mécanique des ressorts. La seconde (lignes 22 à 33) s'occupe de calculer la croissance du tissus et des cellules. Enfin, la dernière fonction (lignes 35 à 39) coordonne les appels des deux transformations de façon à toujours calculer l'état stable du réseau de ressorts et de faire évoluer le tissus.

```
1   fun interaction(c, n) =   // c = cell, n = neighboring cell
2     let dist = distance(c,n)
3     and L0 = c.l0 + n.l0 in
```

```
4         let f = 0.0-K*(dist-L0)/dist in
5             force_to_vector(c, f)
6   ;;
7
8   fun total_interaction(c, n, acc) =
9     acc + interaction(c, n)
10  ;;
11
12  trans Meca =
13  {
14    c =>
15      begin
16        let init = {x=0, y=0, z=0} in
17        let force = fold(total_interfaction(c), init, neighbors(c)) in
18          c + force_to_displacement(force)
19      end
20  };;
21
22  trans GrowthStep =
23  {
24    c / Apical(c) &&  c.age >= seuil =>
25      c + { z = c.z + dz, age = 0 }, // nouvelle cellule apicale
26      subapical1(c), subapical2(c) ;
27    c / Apical(c) =>
28      c + { age = c.age+1 } ;
29    c / Subapical(c) =>
30      vegetative1(c), vegetative2(c), vegetative3(c) ;
31    c / Vegetative(c) && c.l < Maxsize =>
32      c + { l = c.l + sizeinc }
33  };;
34
35  fun meristemGrowth(d) =
36      let s = setify(Meca[‘fixpoint](d)) in
37      let ns = GrowthStep(s) in
38          delaunayfy(graph, ns)
39  ;;
```

Les lignes 1 à 20 permettent de modéliser les interactions mécaniques entre les cellules. La première fonction (lignes 1 à 6) calcule la force exercée par le ressort situé entre la cellule `c` et la cellule `n` sur la cellule `c`. La seconde fonction, utilisée avec la fonction `neighbors` calcule la somme des forces exercées sur une cellule donnée par ses voisines. Enfin, la transformation `Meca` déplace chaque cellule d'un petit déplacement dans la direction de la force. Ici, on peut voir que le calcul du remplacement de la cellule peut être un bloque aussi complexe que l'on veut. Simplement, il faut alors utiliser les mots-clefs `begin` et `end` de MGS.

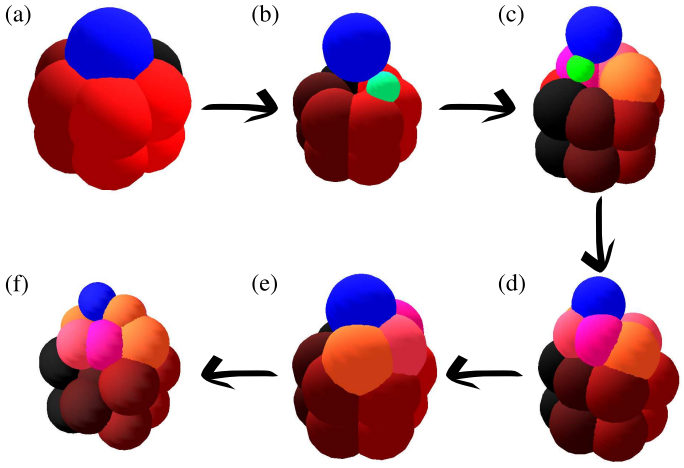


FIG. IV.7: **Modèle de tissus à cellule apicale unique.** Dans cette représentation, chaque cellule est indiquée par une sphère. La cellule apicale est la sphère bleue, les cellules sub-apicales sont de petites sphères vertes et les sphères rouges ou orangées sont les cellules végétatives, leur couleur étant fixée au moment de leur création. **(a)** montre l'état initiale. En **(b)**, la cellule apicale s'est divisée une fois, donnant les cellules sub-apicales représentées en vert. En **(c)**, les cellules sub-apicales se sont divisées (cellules roses/oranges) et la cellule apicale s'est re-divisée. Les images **(d)**, **(e)** et **(f)** montrent comment le tissus évolue sur plusieurs cycles de divisions.

La transformation `GrowthStep`, qui assure la croissance et l'évolution des cellules, est donnée lignes 22 à 33. Dans cette transformation, nous supposons que trois prédicats `Apical`, `Subapical` et `Vegetative` ont été définis pour tester la nature d'une cellule. Ensuite, chaque règle locale se décompose en trois parties et sont séparée par des point-virgules:

```
chemin1 / condition1 => nouveau_chemin1 ;
chemin2 / condition2 => nouveau_chemin2
```

Le chemin est ici à chaque fois réduis à une seule cellule. Par contre, les conditions, si elles portent le plus souvent sur la nature de la cellule, peuvent être légèrement plus complexes. Pour chaque cellule, MGS appliquera la première règle dont la condition est vérifiée et remplacera donc la cellule par le nouveau chemin. Ainsi, une cellule apicale suffisamment âgée pour se diviser le fera, alors qu'une cellule apicale pas assez âgée vieillira simplement. Enfin, les deux dernières règles définissent le comportement des cellules respectivement sub-apicales et végétatives. Si une cellule ne vérifie aucune des conditions, elle est simplement laissée non modifiée.

Enfin, la fonction principale, comme souvent, est placée à la fin du programme (lignes 35 à 39). Elle s'occupe de calculer l'état stable de la mécanique (i.e. calculer l'évolution de la mécanique jusqu'à ce qu'il n'y ait plus de modifications) puis de calculer une étape de croissance. Ainsi, une simulation appellera successivement cette fonction, chaque appel calculant un pas de temps dans l'évolution du tissus. Le résultat d'une simulation de ce système peut être vu sur la figure IV.7.

IV.3 Résultats

Le résultat d'une simulation est obtenu sous la forme d'une série temporelle de graphes représentant l'évolution du méristème virtuel au cours du temps (voir figure IV.8). Au fur et à mesure que les primordia s'éloignent, la tâche d'auxine dans la zone centrale du méristème s'étend selon une direction perpendiculaire à la droite joignant les deux derniers primordia. Lorsqu'ils sont suffisamment loin, elle déborde hors de la zone centrale et provoque l'initiation de deux nouveaux primordia, recommençant le cycle avec une tâche de dimension réduite qui s'étendra à nouveau au fur et à mesure que les primordia s'éloignent.

Nous attendons plusieurs propriétés de notre modèle. Une première propriété est une bonne séparation des primordia. Une deuxième est une croissance uniforme dans le temps et l'espace du tissu. La propriété la plus globale, qui montrerait la cohérence et la complétude* de nos hypothèses serait de reproduire divers modes phyllotaxiques connus. Pour l'instant, nous avons construit un modèle capable de produire des primordia bien séparés, en suivant une phyllotaxie opposée ou opposée décussée, mais nous n'avons pas encore obtenu de phyllotaxie spiralée stable. Notre objectif est donc d'analyser la qualité des résultats obtenus et de déterminer les prochaines étapes du travail.

Nous allons dans un premier temps étudier un certain nombre d'indicateurs du comportement du modèle physique pour déterminer son adéquation aux hypothèses que nous avons formulées. Nous allons ensuite étudier un moyen de tester le modèle physiologique seul.

IV.3.1 Comportement du modèle physique.

Le modèle physique a été construit dans l'hypothèse d'une croissance uniforme dans le temps et l'espace du tissu méristématique. Ceci inclut des divisions cellulaires uniformément réparties, une croissance régulière des cellules et un déplacement radial en moyenne de celles-ci.

Dans cette section, tous les temps seront exprimés en ut (unité de temps) et toutes les distances seront exprimées en ud (unité de distance). Une ut correspond à un pas de calcul mécanique, et est homogène à des secondes. Une cellule a un diamètre moyen de $25ud$, et les ud sont homogènes à des mètres.

IV.3.1.1 Uniformité spatiale de la croissance.

Notre premier test concerne l'hypothèse de croissance uniforme. Si elle est vérifiée, alors une cellule c qui est à une distance r_c du centre c_0 a une vitesse $\vec{V_c}$ donnée par :

$$\vec{V_c} = \alpha r_c \vec{n} \qquad \vec{n} = \frac{\vec{c_0 c}}{c_0 c} \tag{IV.13}$$

Nous avons alors étudié d'une part la vitesse radiale des cellules, et d'autre part la vitesse orthoradiale.

Vitesse radiale. Selon notre hypothèse, la vitesse radiale devrait être proportionnelle à la distance au centre. Si en plus nous supposons un bruit additif* indépendant entre

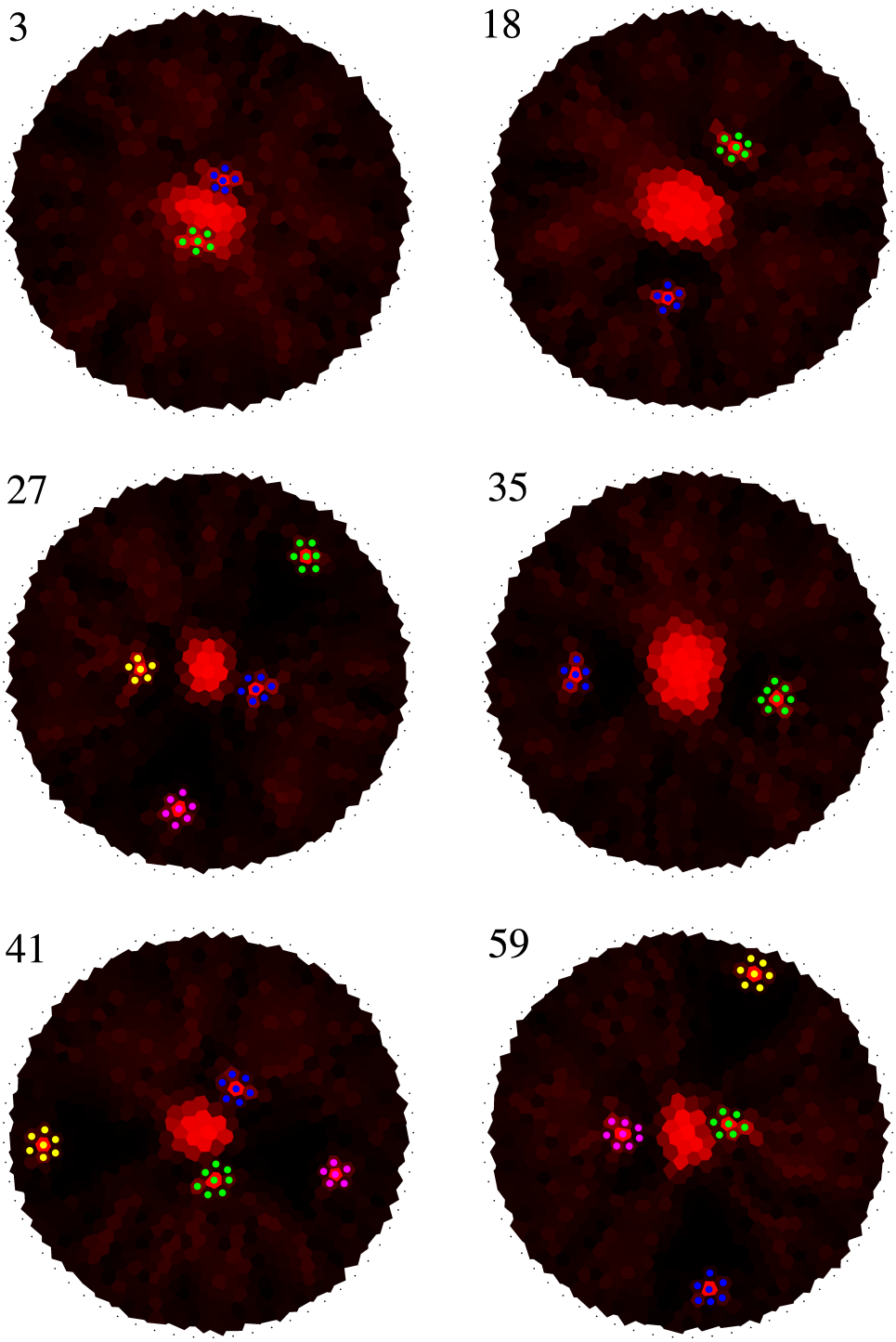


FIG. IV.8: **Résultat d'une simulation** Les cellules des primordia sont marquées d'un point d'une couleur différente pour chaque primordium. Le nombre en haut à gauche de chaque méristème indique le temps écoulé en nombre de cycles complets de calcul.

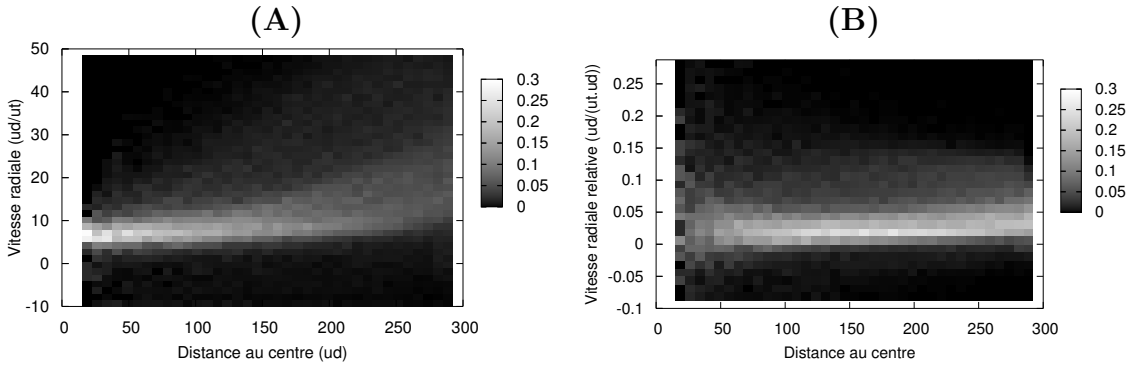


FIG. IV.9: **Vitesse radiale des cellules. (A)** Histogrammes des vitesses radiales des cellules. **(B)** Histogrammes des vitesses radiales des cellules relativement à leur distance au centre.
Chaque colonne des deux figures correspond à l'histogramme pour les cellules se situant à une distance donnée du centre. Ainsi, la somme des fréquences sur chaque colonne est 1.

les cellules, alors le tracé des fréquences d'apparition des vitesses à différentes distances devrait nous montrer un cône s'ouvrant quand la distance augmente (voir figure IV.9A).

La figure montre bien un accroissement de l'étalement et une augmentation de la vitesse moyenne, mais l'augmentation ne semble pas linéaire.

Considérons alors la vitesse relative définie par :

$$\vec{v}_c = \frac{\vec{V}_c}{r_c} \tag{IV.14}$$

D'après IV.13, la vitesse relative devrait être purement radiale, constante et valoir α. La figure IV.9B montre les histogrammes des vitesses relatives radiales pour divers distances au centre. Avec l'hypothèse d'un bruit additif sur la croissance des cellules et en considérant que les cellules ont partout la même taille moyenne, la dispersion de $\vec{v}_c$ devrait rester constante quand le rayon augmente. En effet, la dispersion ne varie pas beaucoup sauf pour des petits rayons, ce qui semble confirmer l'hypothèse d'un bruit additif. La vitesse relative semble bien constante, avec une moyenne sur l'ensemble des données de $0.037 ud/(ut.ud)$ et qui ne varie que très peu si on considère indépendamment les différentes classes de distances au centre. Par contre, si la dispersion est constante, elle est très importante. À la périphérie du méristème, 90% des cellules ont une vitesse radiale contenue dans une plage de $27 ud/ut$, ce qui implique qu'au court d'un seul pas de temps, deux cellules peuvent avoir un déplacement relatif de la taille d'une cellule.

Vitesse orthoradiale. La seconde conséquence de la loi IV.13 est que la vitesse orthoradiale devrait être nulle (voir figure IV.10). La vitesse orthoradiale moyenne est bien nulle quelle que soit la distance au centre. Par contre, il est étonnant de voir que la variabilité est constante. Ainsi, l'écart-type de la vitesse orthoradiale, qui est de $3.6 ud/ut$, est déjà très élevé pour la périphérie où la vitesse radiale moyenne est de $7.5 ud/ut$ mais pour les points proches du centre où la vitesse moyenne est de $1.25 ud/ut$, il n'est plus possible de considérer qu'ils ont une vitesse radiale, même approximativement.

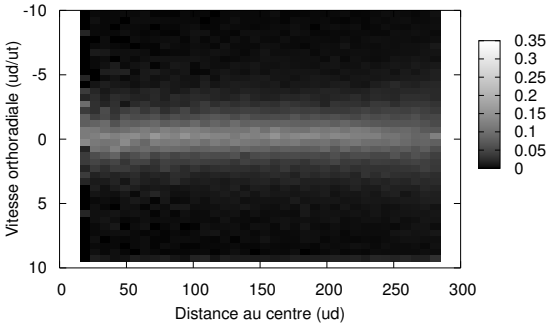


FIG. IV.10: **Vitesse orthoradiale des cellules.** Il est intéressant de noter que la vitesse orthoradiale moyenne est nulle et que, en première approximation, la dispersion semble régulière quelle que soit la distance au centre.

Division cellulaire. Si la croissance est uniforme, comme notre modèle de division cellulaire ne dépend que de la taille de la cellule, alors elle devrait elle aussi être uniforme. Toutefois, on remarque que le nombre de divisions cellulaires est plus important en périphérie qu'au centre (voir figure IV.11). Il est donc très probable que la croissance cellulaire soit plus importante en périphérie qu'au centre.

Il est intéressant de séparer les divisions qui ont lieu au cours d'une même simulation selon que le nombre total de divisions cellulaires pour une seule étape de calcul est petit ou grand (le seuil est ici de 80 divisions, voir section IV.3.1.2). On remarque alors que la différence entre périphérie et centre est bien plus marqué quand il y a peu de divisions, et pratiquement absente quand il y a eu beaucoup de divisions.

IV.3.1.2 Uniformité temporelle de la croissance.

Enfin, le dernier test réalisé sur le comportement du modèle mécanique a porté sur l'uniformité temporelle de la croissance.

Comme nous l'avons vu dans la section précédente, la vitesse relative est constante en moyenne, quelle que soit la position de la cellule. Nous pouvons donc l'utiliser comme indicateur global de la vitesse de croissance moyenne sans introduire de biais sur la position des cellules. C'est une information importante, car il y a beaucoup plus de cellules loin du centre que proche du centre. Nous avons aussi regardé le nombre total de divisions dans le méristème.

La figure IV.12A nous montre l'évolution temporelle de ces deux grandeurs. L'amplitude du mouvement étant d'un facteur 14, il n'est pas possible de parler d'uniformité temporelle. Par contre, il est très net que les pics de croissances sont suivis, au temps d'après, par un pic de divisions cellulaires. Aussi, il est intéressant d'étudier la corrélation entre la vitesse moyenne d'éloignement au temps t et le nombre de divisions au temps $t+1$. La figure IV.12B montre cette corrélation. De toute évidence, les grandes vitesses et les grands nombres de divisions sont très fortement corrélés, alors que les petites valeurs beaucoup moins. Il est même possible de déterminer la limite aux alentours de 80 divisions et $0.08ud/(ut.ud)$. Il semble que, au-dessous de $0.08ud/(ut.ud)$, le nombre de divisions soit compris entre un minimum proportionnel à la vitesse et 80. Il semblerait

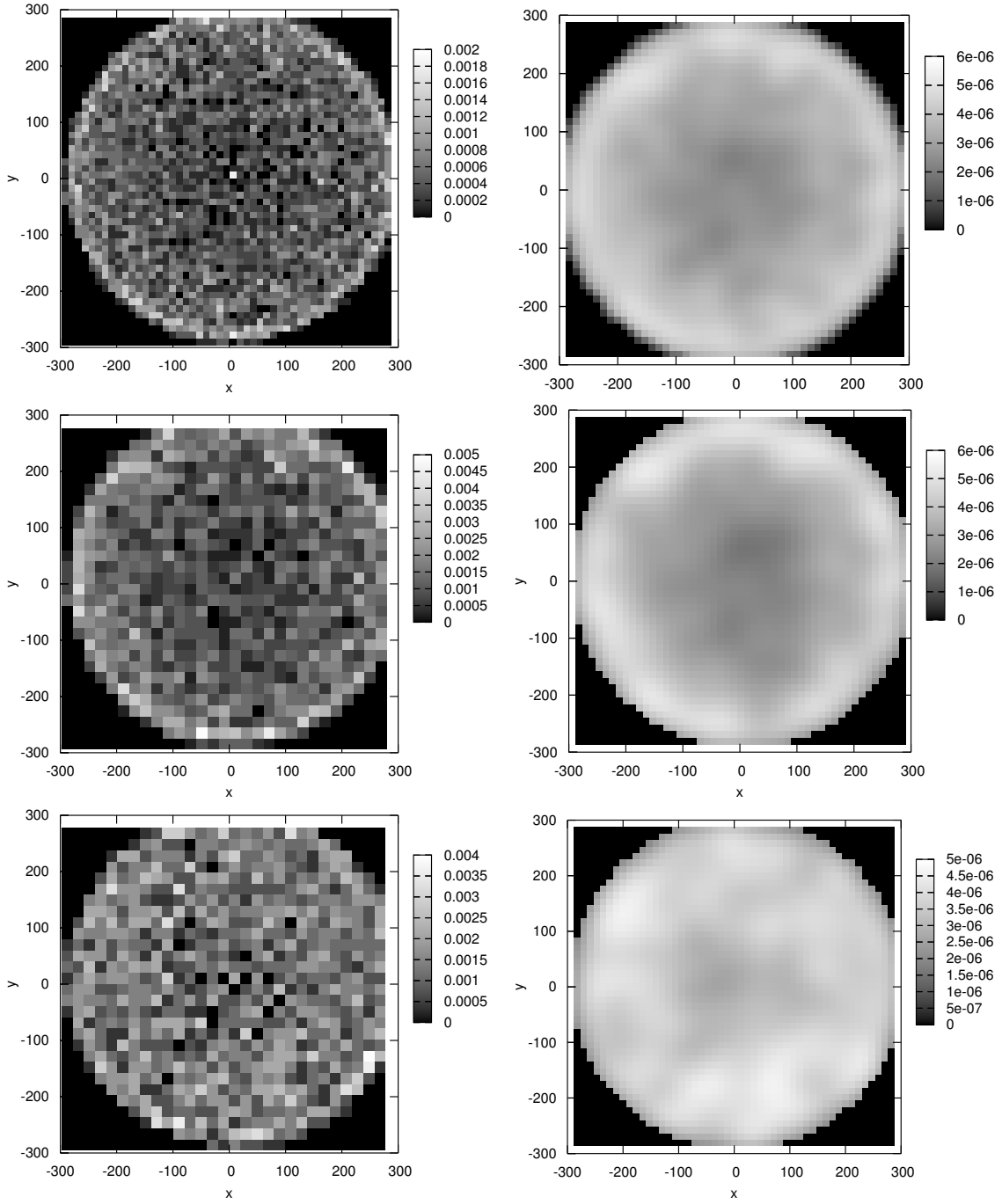


FIG. IV.11: **Position des divisions cellulaires.** La première ligne montre la distribution des divisions cellulaires lors d'une simulation. Le modèle produit plus de divisions à la périphérie du méristème. La deuxième ligne montre la même distribution mais uniquement pour les étapes de calcul où il y a eu moins de 80 divisions cellulaires, où la différence entre le centre et le bord est plus accentuée. Enfin, la troisième ligne correspond aux étapes de calcul où il y a eu plus de 80 divisions cellulaires où la distribution des divisions cellulaires est quasi-uniforme.
La colonne de droite correspond à l'estimation de la densité de probabilité (i.e. un lissage avec une fonction gaussienne) des divisions cellulaires sur la surface du méristème virtuel.

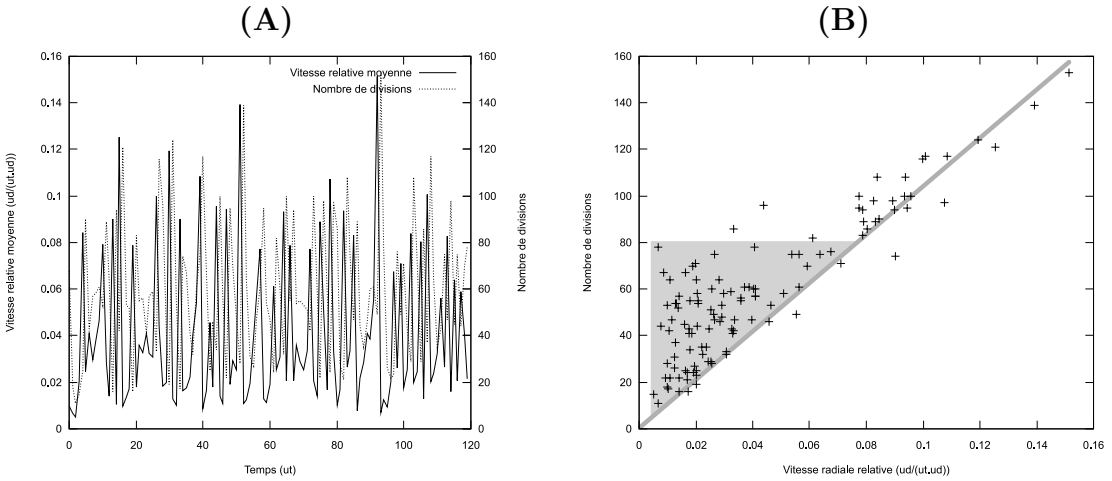


FIG. IV.12: **Répartition temporelle des vitesses et des divisions cellulaires.** **(A)** Évolution temporelle du nombre de divisions et de la vitesse relative d'éloignement des cellules. **(B)** Relation entre la vitesse d'éloignement des cellules au temps t et le nombre de divisions au temps $t+1$. En gris sont représentées les deux classes de points : sur la droite les points pour lesquels le nombre de divisions est proportionnel à la vitesse relative, et dans le triangle les points pour lesquels il y a plus de divisions que prévu.

que les divisions "supplémentaires" aient lieu à la périphérie du méristème et soient dues à un effet de bord. Cette impression est renforcée par la constatation qu'une vitesse radiale relative de $0.08 ud/(ut.ud)$ correspond à une vitesse radiale à la périphérie du méristème d'environ $24 ud/ut$, soit à peu près une cellule par pas de calcul. Donc les cellules qui sont à l'extérieur au pas de temps précédent ont presque toutes disparues et ne sont pas considérées dans le calcul des divisions cellulaires. Cette disparition implique la suppression de l'effet de bord qui, s'il se produit effectivement, n'est simplement plus pris en compte dans les analyses.

IV.3.2 Comportement du modèle physiologique.

Pour tester le modèle physiologique indépendemment du modèle physique, nous allons utilisé un modèle macroscopique reconnu pour sa robustesse : le modèle des champs de répulsions décrit en section I.2.2 et détaillé par Douady et Couder (1996a).

Ce que nous retiendrons de ce modèle n'est pas le choix de l'instant où un primordium doit être créé, car c'est fortement dépendant de la façon dont les primordia s'éloignent et donc de la mécanique. Nous ne retiendrons que la règle qui permet de placer le primordium à un moment donné. Chaque primordium émet un champ de répulsion qui décroît avec le carré de la distance au primordium. La formation d'un primordium a lieu sur un cercle

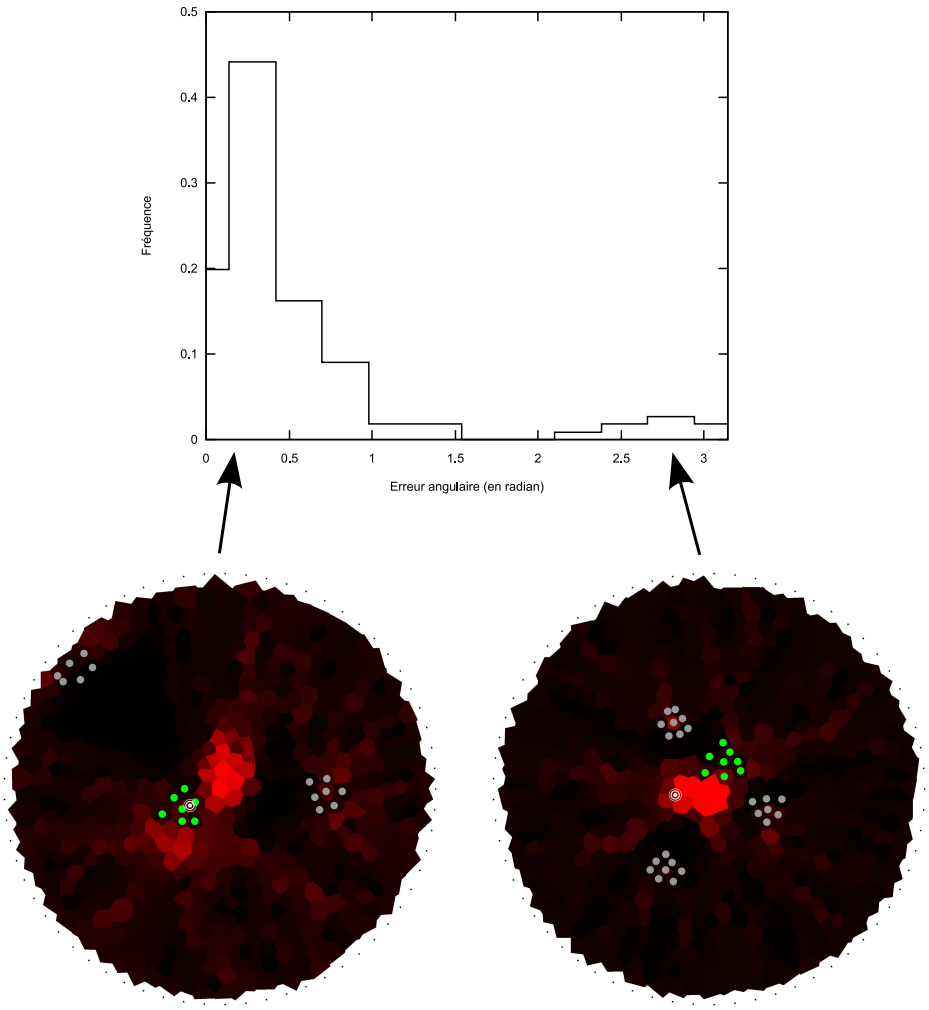


FIG. IV.13: **Erreur de positionnement angulaire des primordia.** Le graphique présente l'histogramme des fréquences d'apparition des erreurs. La largeur des colonnes a été choisie de façon à représenter en moyenne une cellule. Ainsi, on peut voir que 20% des primordia sont bien placés et que plus de 60% des primordia sont placés au pire dans une cellule voisine de la cellule correcte et 92% sont à moins de 1 radian (et donc dans le bon secteur). Le méristème de gauche montre une étape de simulation ou le nouveau primordium (en vert) qui vient d'apparaître a été correctement placé (cercles blancs concentriques), alors que dans celui de droite, le nouveau primordium a été placé au second minimum d'énergie.

de rayon r_M au point d'angle $\widehat{\theta}$ où le champ de répulsion est minimum :

$$E(x) = \sum_{p \in \widetilde{\mathcal{P}r}} \frac{1}{d(x,p)^2} \quad \text{(IV.15)}$$

$$\widehat{\theta} = \arg\min_{\theta \in [0;2\pi]} E((r_M, \theta)) \quad \text{(IV.16)}$$

où $\widetilde{\mathcal{P}r}$ est l'ensemble des primordia, et (r_M, θ) est le point de rayon r_M et d'angle θ.

Nous définissons alors l'erreur de positionnement par rapport à ce modèle de référence (voir figure IV.13). Il est intéressant de noter que plus de 60% des primordia sont placés correctement (i.e. avec au plus une cellule d'écart avec la position de référence). Par ailleurs, on observe un second maximum loin de la position idéale. Il s'agit le plus souvent du second minimum d'énergie sur le cercle d'apparition des primordia.

IV.4 Analyse de l'utilisation de MGS.

Dans le choix d'un outil pour la modélisation, il y a plusieurs critères à prendre en compte. Le plus important est la capacité à exprimer les hypothèses du modèle dans le langage sans circonvolutions (i.e. sans avoir à transformer de manière équivalentes mais plus complexes des hypothèses sans pour autant apporter un gain significatif en vitesse ou en stabilité numérique). La facilité d'expression est aussi un point important en ce qu'elle conditionnera la vitesse de développement et donc la facilité que nous aurons pour tester différents jeux d'hypothèses. La gestion du calcul numérique peut devenir un problème important, à la fois pour la vitesse de calcul et pour les problèmes de stabilité numérique, en particulier pour la résolution des systèmes d'équations différentielles. Enfin, les outils d'aide au développement, comme la pertinence des messages d'erreurs ou l'existence d'outils de debuggage, sont à ne pas négliger car ils conditionnent aussi fortement le temps de développement du modèle.

Capacité d'expression. Le langage MGS permet d'exprimer directement la plupart des hypothèses que nous avons prises. En fait, il permet même d'exprimer des processus bien plus complexe ce dont nous avons besoin, comme par exemple les interactions à plus de deux éléments, voir à nombre variable d'éléments.

Seul un problème d'implémentation (et non de spécification) ne permet pas d'exprimer directement la division cellulaire. Le problème vient de la structure des graphes de Delaunay. Celle-ci fait partie des collections "newtonienne" de MGS dans lesquelles l'espace étant pré-existant il ne peut être modifié par une transformation. Ainsi, il n'est pas possible de remplacer une cellule par deux cellules lors d'une transformation. Le problème peut être contourner en deux transformations. La première, sur le graphe de Delaunay, calcul quelles cellules doivent se diviser. Avant d'appliquer la seconde transformation, la structure de Delaunay est oubliée pour ne garder que l'ensemble des sites. Un ensemble étant une structure leibnizienne, nous pouvons appliquer une transformation réaliser la division cellulaire dessus. À la fin, la structure de Delaunay est recalculée à partir du nouvel ensemble de sites. Encore une fois, cette limitation est simplement due à l'implémentation et devrait être levée dans les prochaines version de MGS.

Un problème plus important est lié aux perspectives d'évolution de notre modèle. En effet, nous envisageons de passer d'une structure de graphe de Delaunay à une structure de graphe topologique plus générale. Or, pour le moment, MGS ne permet pas d'exprimer l'évolution structurelle d'un graphe car le remplacement d'un chemin ne suffit pas à la spécifier et il est nécessaire de faire appel à la notion de couture, qui est bien plus complexe à exprimer dans le cas général.

Facilité d'expression. MGS propose d'exprimer les transformations de structures topologiques complexes par des règles locales uniquement. C'est une méthode propre aux systèmes de réécritures en général. Or les données dont nous disposons sont aussi exprimées localement en terme d'interaction entre cellules voisines ou en terme d'évolution d'une cellule seule. Ainsi, il est aisé d'exprimer les données dont nous disposons. De plus, la gestion des cas particuliers ou des conditions aux limites se fait très simplement en rajoutant des règles locales et sans "polluer" l'expression du cas général.

Résolution numérique. Dans notre modèle, nous avons eu à résoudre des systèmes d'équations différentielles ordinaires (ODE, *Ordinary Differential Equations*). Or les méthodes efficaces de résolution de tels systèmes reposent essentiellement sur le calcul matriciel ou sur des optimisations globales (pour revue : Press *et al.*, 1992, chap. 16), ce qu'il n'est pas possible de faire efficacement avec un langage comme MGS. Des recherches ont déjà était menées pour améliorer l'efficacité de la résolution des ODE dans le cadre des L-système (Federl et Prusinkiewicz, 2004), mais leur méthode repose pour beaucoup sur la notion d'ordre (ou pré-ordre) existant dans les séquences et les arbres et n'est donc pas applicable, sans recherches supplémentaires, au problème plus général de la résolution sur un graphe. Ainsi, nous sommes limités à une résolution utilisant la méthode d'Euler, qui d'une part ne garantit pas la convergence numérique des équations à moins de diminuer suffisamment le pas utilisé pour la résolution, et donc d'augmenter considérablement le temps de résolution. Pour notre problème, le temps de résolution des ODE s'est avéré critique, et il nous semble donc important de réfléchir à une meilleure technique de résolution.

Outils d'aide au développement. Étant donné le caractère expérimental de MGS, les outils d'aide au développement sont de bonne qualité. Les messages d'erreurs nécessitent de comprendre un minimum la manière dont le langage fonctionne, mais cela ne nécessite qu'un peu de travail. Le problème majeur est le manque de localisation des erreurs. Dans les implémentations actuelles, seule la fonction la plus interne où se déroule l'erreur est connue. C'est le plus souvent insuffisant pour déterminer le lieu de l'erreur. D'une manière générale, les outils d'aide au développement disponibles pour un langage expérimental et à public restreint sont bien moins développés quand pour des langages dont la communauté est extrêmement grande comme Python, Ruby, Java, C++.

Un autre point important dans le développement est le support de l'équipe de développement ou de la communauté. Même si l'équipe de développement de MGS est relativement réduite, elle est très réactive pour répondre aux questions et pour aider quand des problèmes se présentent.

IV.5 Conclusion

La construction de ce modèle nous a permis d'évaluer la pertinence de plusieurs choix. D'une part nous avons pu exprimer assez simplement les connaissances biologiques sur une structure cellulaire. D'autre part, nous avons pu tester un modèle de mécanique cellulaire très simple et ainsi dégager certaines propriétés qui semblent nécessaires à l'établissement d'une phyllotaxie correcte.

Le modèle mécanique que nous proposons permet de simuler une croissance du tissus par la croissance locale des cellules et l'insertion de nouvelles cellules par division cellulaire. Ainsi, nous avons réussit à exprimer l'évolution physiologique des cellules sans être intimement lié à la structure locale du complexe cellulaire. Ensuite, même si le déplacement n'est pas le déplacement idéal tel que modélisé dans les modèles décrit en section I.2.2, il respecte en moyenne les contraintes que nous avions fixées. Toutefois, le bruit semble trop important pour que la combinaison de ce modèle mécanique et le modèle physiologique produise un phyllotaxie correcte. Il s'avère donc nécessaire de travailler sur la nature de ce bruit et sur les moyens de le réduire suffisamment pour obtenir une phyllotaxie correcte.

Quant au modèle physiologique, il donne des résultats très satisfaisant. Ces résultats encourageants nous invitent à étudier plus en détail les propriétés de ce modèle pour éventuellement l'améliorer, et surtout pour déterminer les contraintes qu'il impose au modèle mécanique pour pouvoir fonctionner correctement.

Chapitre V

Implémentation

Pendant cette thèse, deux logiciels ont été intégralement conçus pour intégrer les différentes techniques informatiques décrites dans cet ouvrage. Le premier est appelé Merryproj et implémente les techniques décrites au chapitre II pour la reconstruction de l'image du méristème vue de dessus. Le second est appelé Merrysim et implémente les autres techniques originales de cette thèse.

V.1 Le projet ALEA

ALEA est un projet ayant pour objectif la création d'un atelier logiciel pour l'étude et la modélisation des plantes (Pradal *et al.*, 2004). C'est d'abord le REA (Réseau Écophysiologie de l'Arbre) de l'INRA qui a financé le projet, puis l'INRIA par l'intermédiaire du projet Virtual Plants.

ALEA a pour ambition majeure de fournir un cadre de développement et d'exécution pour développer des modèles de plante. Ceci implique la définition d'interfaces pour les structures de données communes, l'existence d'un bus logiciel pour faire communiquer les composants et enfin une interface graphique homogène pour l'utilisateur. ALEA a aussi pour but de faciliter l'accès aux outils informatique pour des biologistes en leur permettant notamment de construire leurs modèles en assemblant les modèles et les algorithmes existant.

En tant que projet destiné à intégrer la plate-forme ALEA et étant au cœur de l'équipe développant la plate-forme, Merrysim et Merryproj ont servi de plate-forme de test pour un certain nombre de techniques de développement. Aussi, certains choix s'expliquent dans le cadre plus large de ce projet et non dans le cadre plus restreint de cette thèse.

V.2 Merrysim

Merrysim est le logiciel principal développé pendant cette thèse. Il intègre :
- un ensemble d'interfaces graphiques de saisies ;
- un interpréteur permettant d'accéder interactivement aux objets saisis ou en cours de saisie, de les modifier et les analyser ;
- des structures de données dédiées ;

– des algorithmes d'extraction et d'analyse de données ;
– des outils de représentation des données;
– un moteur de simulation.

En plus de ces outils, Merrysim permet d'accéder à de nombreuses bibliothèques développées pour le langage Python, soit des bibliothèques développées au sein de l'équipe Virtual Plants (PlantGL, amlPy), soit dans la communauté Python (SciPy, Gnuplot, ...).

Le résultat est une plate-forme logicielle dédiée à l'étude du méristème, utilisant les possibilités du langage Python pour faire interagir des bibliothèques développées par des équipes différentes dans des langages différents.

Par rapport au projet ALEA, Merrysim est bien moins ambitieux. Il n'y a pas eu de recherche menée pour facilité l'intégration des bibliothèques diverses au-delà de ce que propose le langage Python, il n'y a pas eu de problèmes de structures de données à multiples implémentations, et enfin il n'y a qu'un seul développeur pour l'application principale, ce qui évite les problèmes de développements concurrents. Toutefois, puisque d'une part Merrysim est destiné à fusionner avec la plate-forme ALEA et d'autre part il a été développé au sein de l'équipe en charge d'ALEA, son développement a permit de tester des outils et des techniques qui seront utiles à la plate-forme ALEA.

V.2.1 Architecture générale

Au cœur de l'application Merrysim, le noyau est formé d'un ensemble de structures de données et d'algorithmes écrits en C++ (voir figure V.1). Il est rendu accessible depuis le langage Python par l'intermédiaire d'une extension au langage Python écrite à l'aide de la bibliothèque Boost.Python (Abrahams, 2003).

Pour faciliter la construction des structures de données à partir des données biologiques, Merrysim propose une interface homme-machine (IHM) graphique. Elle se compose d'un ensemble d'interfaces de saisie basées sur les structures et algorithmes disponibles dans le noyau (voir section V.2.2). L'IHM permet d'éditer simultanément plusieurs objets et de les lire/écrire dans des fichiers. Elle est aussi prévue pour être facilement extensible en C++ en proposant notamment un objet générique d'affichage de tissus cellulaire 2D. L'IHM embarque aussi un interpréteur Python permettant d'accéder directement d'une part aux objets en cours d'édition, et d'autre part à l'interface elle-même.

V.2.2 Interfaces de saisie

Les interfaces de saisies fonctionnent toutes sur le même principe. La saisie est réalisée à partir d'un objet observé non modifiable. L'objet de travail est construit au fur et à mesure de la saisie en surimpression sur l'objet observé. Si l'objet observé est modifié par ailleurs, il faut recharger l'objet de travail qui conservera autant que possible ce qui a été fait mais qui détruira les informations devenues inexploitables (par exemple, si une cellule est supprimée, les liens créés depuis ou vers cette cellule seront supprimés aussi au chargement).

Sur ce principe, cinq interfaces de saisies ont été réalisées :

1. saisie des cellules du méristème par les paroi (objet observé : image du méristème) (voir figure V.2)

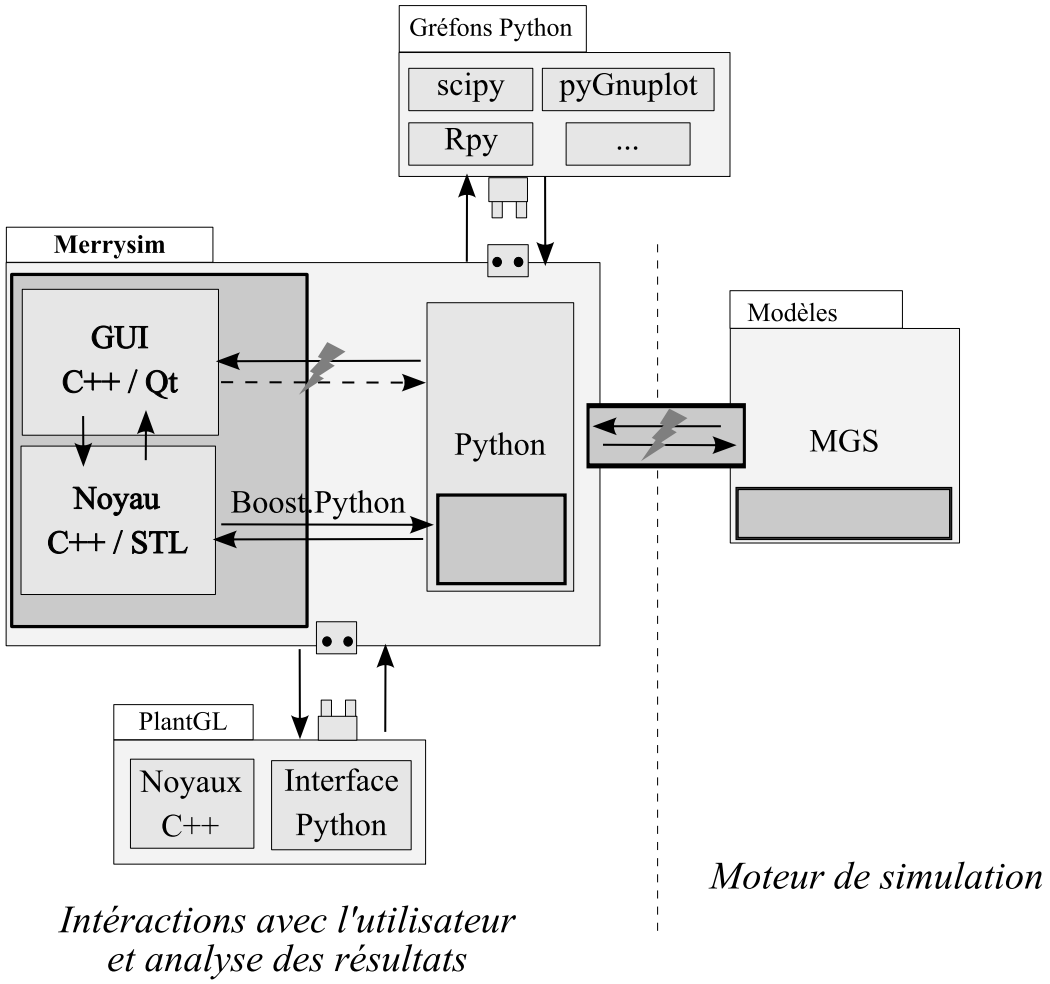


FIG. V.1: **Architecture globale de Merrysim**

Merrysim est construit de façon a pouvoir interagir d'une part avec n'importe quel module accessible depuis un interpréteur Python standard et d'autre part avec des scripts MGS pourvu que les objets créés puissent être envoyés à Python (voir section V.3).

La ligne en pointillé sépare deux programmes différents, ce qui implique un système de communication inter-processus pour traverser cette ligne et un système intra-processus pour les autres systèmes de communication.

Les travaux réalisés spécifiquement pendant ce projet sont indiqués par des cadres en traits épais, les éclairs indiquant les éléments particulièrement délicats à réaliser.

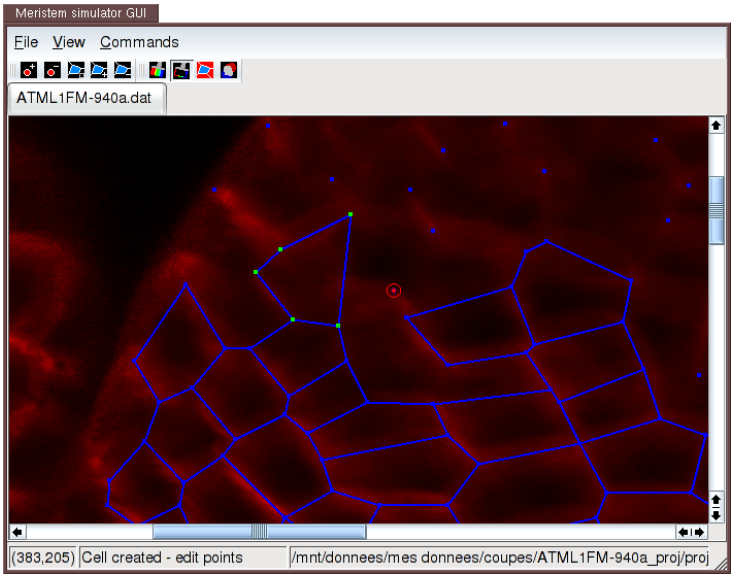


FIG. V.2: **Interface de saisie des parois cellulaires.** Cette interface permet de définir les vertex des cellules (i.e. les points communs à trois cellules), et de les grouper par cellules. La forme de la cellule est automatiquement déduite de l'ensemble non-ordonné des vertex (voir chapitre II).

2. saisie des cellules du méristème par les centroïdes (objet observé : image du méristème) (voir figure V.3)
3. saisie des cellules sœurs dans le méristème (objet observé : méristème digitalisé et image du méristème) (voir figure V.4)
4. liens entre les cellules (objet observé : méristème digitalisé et image du méristème) (voir figure V.5)
5. filiations cellulaires (objet observé : vues d'un méristème à deux instants successifs) (voir figure V.6)

Les interfaces 2 à 5 sont toutes basées sur un même objet permettant de représenter un tissus cellulaire en 2D (cet objet est présent deux fois pour la filiation cellulaire: une fois pour chaque méristème).

En plus de présenter un tissus cellulaire 2D, cet objet notifie le système des actions de l'utilisateur sur sa fenêtre. En particulier, les actions prévues sont : clique, passage sur une cellule ou un vertex, déplacement de la souris, déplacement d'un centre de cellule, clique sur un point vide. En fonction de l'interface, le traitement réalisé sur ces actions n'est pas le même. La déconnexion entre l'affichage et les réactions aux actions de l'utilisateur a permis de réutiliser un unique objet dans 4 interfaces travaillant sur des objets différents tout en gardant un maximum de cohérence à travers ces interfaces. Par exemple, la validation se fait toujours de la même façon, par un clique droit, car c'est une abstraction proposée par l'objet de visualisation, mais ce que fait une validation est différent dans chaque cas.

Même si les données recueillies par les différentes interfaces ne sont pas les mêmes, elles reposent toutes sur une structure de donnée commune permettant de représenter les

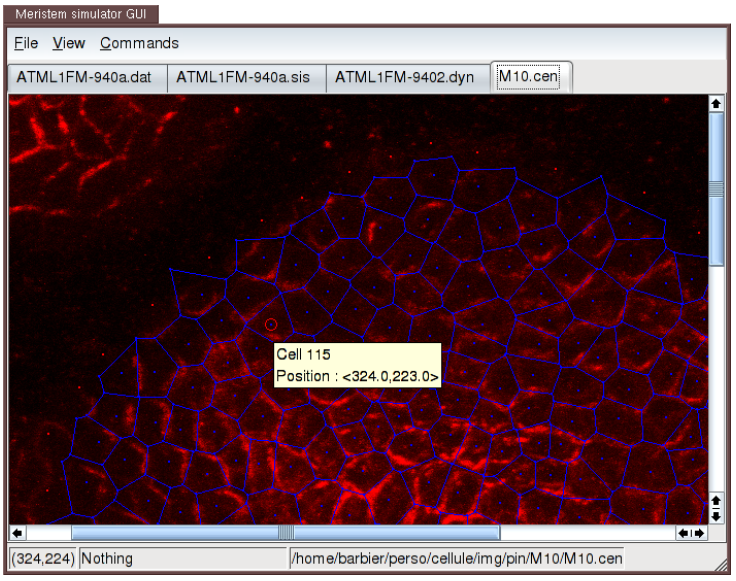


FIG. V.3: **Interface de saisie des cellules par leurs centroïdes.** Cette interface permet de définir les cellules en spécifiant les centroïdes qui serviront à calculer les domaines de Dirichlet et le graphe de Delaunay associé (voir section IV.1.1). Les points rouges sont les centroïdes des ancres.

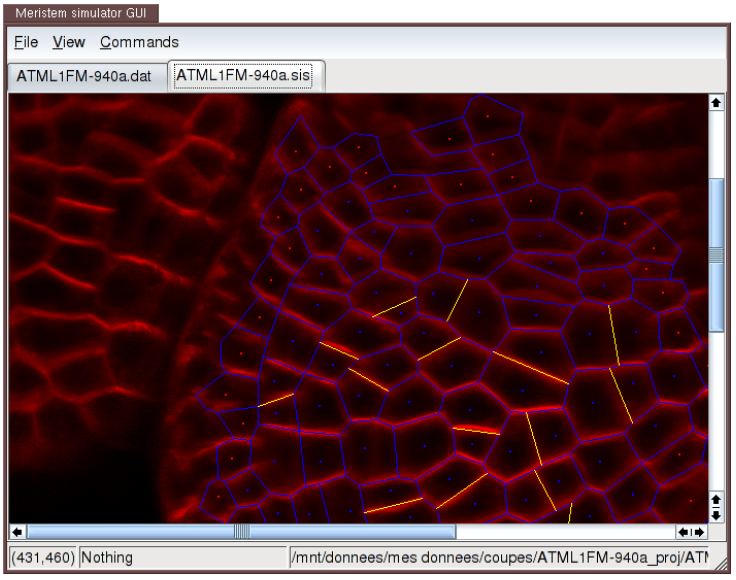


FIG. V.4: **Interface de saisie des cellules soeurs** Les parois dessinées en jaunes séparent deux cellules sœurs.

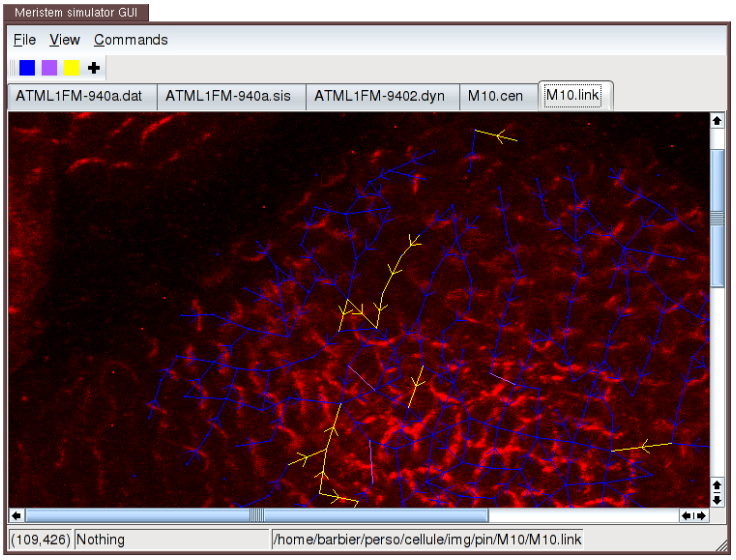


FIG. V.5: **Interface de saisie des liens entre cellules voisines.** Cette interface permet de saisir des liens de nature et de sémantique différentes (comme par exemple, les différents niveaux de confiances accordés au positionnement de la protéine PIN1). Il ne peut exister qu'un seul arc ayant même source et destination, quel que soit le type de lien.

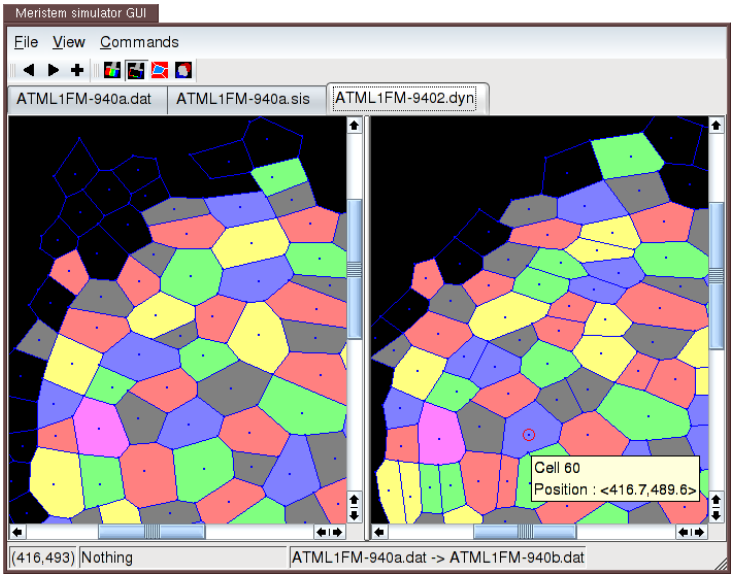


FIG. V.6: **Interface de saisie de la filiation cellulaire.** Dans cette interface, la fenêtre de gauche montre un méristème au temps t et la fenêtre de droit le même méristème au temps $t+dt$. Les cellules au temps $t+dt$ ayant même mère au temps t sont représentées avec la même couleur.

tissus cellulaires en 2D. Nous allons donc maintenant expliquer le fonctionnement de ces objets.

Extensibilité et modularité des interfaces de saisies. Par rapport à leur conception, les interfaces de saisies peuvent être scindées en trois groupes :

1. la saisie des cellules sœurs, des liens entre cellules et des méristèmes par les centroïdes ;
2. la saisie des lignées cellulaire ;
3. la saisie d'un méristème par les parois.

Les interfaces du premier groupe sont toutes basées sur une même classe `Meristem` qui permet de représenter un méristème et qui est conçue pour être facilement étendue par dérivation. Cette classe est un widget* Qt et n'est que très peu liée au reste de l'interface graphique de Merrysim. L'ajout d'une interface dans cette catégorie est une opération relativement simple. Il faut d'abord créer une classe fille de `Meristem` et redéfinir quelques méthodes utilitaires dont les méthodes d'affichage. Il y a deux méthodes d'affichage : la première est celle standard dans Qt (`paint_event`) et sert à dessiner les objets mobiles ou temporaires directement dans le widget ; la seconde s'appelle `Redraw` et sert à dessiner les éléments persistant du dessin dans un buffer. Ces deux méthodes sont déjà implémentées dans la classe `Meristem` pour l'affichage du méristème lui-même ainsi que pour les outils de sélection. Ensuite, il est nécessaire de modifier la classe `EditSingle` qui permet de regrouper tous les éditeurs de cette catégorie. Une seule méthode de cette classe est à modifier : cette méthode (`load_meristem`) est la seule à dépendre effectivement de la liste des classes existantes. Enfin, il faut rajouter les éléments d'interface dans le menu de création d'un nouvel objet et dans la liste des actions pour charger un objet existant. Pour une interface relativement simple comme celle permettant de saisir les cellules sœur, la création de l'interface graphique implique l'écriture d'environ 300 lignes de code C++ et a nécessité environ une journée de travail (il avait fallu deux jours en comptant la création des structures de données, du format de sauvegarde et les tests de l'ensemble). Il faut toutefois noter que l'ajout d'une nouvelle classe nécessite d'intervenir dans du code existant, même s'il est bien circonscrit. Ainsi, il n'est pas possible par ce moyen de rajouter dynamiquement des interfaces de saisies. Quant à l'extraction des classes étendant `Meristem`, elle est relativement aisé car la partie de code liée à Merrysim est bien circonscrite. Toutefois, cela demandera plus de travail que pour la classe `Meristem` elle-même.

Les deux autres groupes sont des interfaces *ad hoc*, même si l'interface de saisie des lignées cellulaires est aussi basée sur la classe `Meristem` instanciée deux fois : une fois par représentation de méristème. Ces classes sont bien plus dépendantes de l'interface graphique de Merrysim et l'extraction sera probablement plus difficile que pour les interfaces du premier groupe.

V.2.3 Le noyau

Les objets principaux du noyau sont les graphes et les graphes cellulaires. Aussi nous commencerons par les décrire. Ensuite, nous verrons un certain nombre d'objets et d'algorithmes présents dans le noyau.

V.2.3.1 Graphes

La classe de graphe est la pierre angulaire de l'application Merrysim. C'est la structure de donnée la plus utilisée sur laquelle repose toutes les interfaces de saisie et pratiquement tous les algorithmes.

Pour la conception de la bibliothèque de graphe les contraintes étaient :
- édition des graphes en temps constant (ajout/suppression des arcs et des nœuds) ;
- taille mémoire raisonnable ;
- possibilité d'associer plusieurs classes d'arcs à un même ensemble de nœuds ;
- facilité de création de types d'arcs ;
- interface Python ;
- extensible en C++ ;
- respect des interfaces proposées par le projet ALEA (voir annexe A) et la Boost Graph Library.

Pour permettre plusieurs classes d'arcs (i.e. plusieurs graphes utilisant le même ensemble de vertex), chaque classe d'arc ayant des propriétés différentes, l'objet principal (une instance de la classe `Graph`) maintient la liste des classes d'arcs dont il a besoin (des instances de la classe `GraphLink`). Par la suite, l'objet principal fonctionne par délégation* pour les fonctions qui font intervenir les arcs. La désignation de l'objet de destination peut être faite par deux moyens: soit en spécifiant le dernier argument optionnel qui est toujours l'identifiant de la classe d'arc à utiliser, soit en spécifiant la classe d'arc courante avant l'appel des fonctions. L'existence d'une classe d'arc courante spécifiable est indispensable pour une compatibilité avec les interfaces de ALEA et de la BGL car elles ne prévoient à aucun moment l'existence de plusieurs classes d'arcs.

Une des difficulté était de permettre à la fois la création de nouveaux types d'arcs en évitant la redondance de code et l'exportation en Python. Une approche en C++ pur aurait consisté en l'utilisation de patrons* de classes instanciés pour les différents types d'arcs. Mais les patrons de classe ne sont pas exportable en Python (seul les patrons instanciés le sont). Nous avons alors choisi d'utiliser la dérivation et le polymorphisme* pour découpler l'interface de l'implémentation. Le polymorphisme est lui aussi problématique car il impose l'usage de pointeurs ou de références. D'une part les références ne sont pas exportable en Python, et d'autre part les pointeurs ne sont pas pratique d'utilisation (il est nécessaire de les déréférencer pour les utiliser). Nous avons donc choisi d'utiliser des proxy* pour utiliser des pointeurs tout en donnant l'illusion d'utiliser directement l'objet lui-même.

Cette abstraction a permis d'implémenter six classes d'arcs ayant des propriétés différentes :

1. des arcs dirigés : classe `DirectedLink`
2. des arcs non-dirigés : classe `UndirectedLink`
3. des arcs dirigés sans doublon : classe `UniqDirectedLink`
4. des arcs non-dirigés sans doublon : classe `UniqUndirectedLink`
5. des arcs construisant le graphe de Delaunay : classe `DelaunayLink`
6. des arcs construisant le co-graphe d'une autre classe d'arc : classe `CographLink`

Extensibilité des classes de graphe. Pour notre classe de graphe, l'extensibilité concerne la création d'un nouveau type d'arc. La structure de la bibliothèque est telle qu'il suffit de dériver de la classe `GraphLink` qui définit l'ensemble des méthodes qui peuvent être implémentées dans une classe d'arc. Toutes les méthodes ont une implémentation par défaut qui lève une exception indiquant qu'elle n'est pas implémentée dans les classes de base. Ainsi, il suffit d'implémenter les méthodes pertinentes sans se préoccuper des autres méthodes (par exemple, une classe de graphe non-modifiable n'implémentera simplement pas les méthodes d'ajout et suppression d'arcs). L'utilisation de proxy* et de polymorphisme* permet d'étendre les classes d'arc dynamiquement. Ainsi, si un module d'extension Python implémente une nouvelle classe d'arc, il suffit de le charger et la classe est utilisable à l'exécution. Inversement, aucune implémentation n'est nécessaire à l'utilisation de la bibliothèque. Dans un cas extrême, il serait possible de ne proposer aucune classe d'arc et de laisser l'utilisateur charger, par l'intermédiaire de modules, celles dont il a besoin.

Gestion des propriétés sur les graphes. La gestion des propriétés sur une structure est une opération complexe. La Boost Graph Library propose une solution à travers son concept* de "Property Graph". Ce concept repose lui-même sur une autre bibliothèque du projet Boost, la Boost Property Map Library (BPML ; Siek *et al.*, 2001, chap. 15) qui a été extrait de la Boost Graph Library. Les propriétés définies par cette bibliothèque reposent sur deux fonctions `get` et `put`, respectivement pour récupérer et écrire une valeur associée à un élément de la structure. Toutefois, la difficulté du problème n'est pas la création, la modification et la récupération de valeurs mais la suppression. Si la structure est dynamique et les données attachées aux éléments volumineuses, il devient prépondérant de pouvoir effacer les données en même temps que les éléments. C'est un problème qui n'a malheureusement pas été adressé par la BPML, aussi nous ne l'avons pas utilisée pour gérer les propriétés de nos graphes. Toutefois, la conception de la BPML permet d'utiliser les algorithmes reposant dessus pourvu que les fonctions `get` et `put` puissent être fournit, ce qui devrait toujours être le cas pour des propriétés.

L'implémentation des graphes de Merrysim ne propose pas de gestion centralisée des propriétés. La solution retenue est de reléguer la gestion des propriétés aux classes filles qui connaîtront la liste des propriétés à maintenir et auront la charge de redéfinir les méthodes d'édition des nœuds et des arcs pour mettre à jour les propriétés.

V.2.3.2 Graphes cellulaires

Comme nous l'avons vu, la bibliothèque de graphe ne propose pas de gestion centralisée des propriétés. Les graphes cellulaires sont des graphes augmentés de propriétés sur les nœuds, à savoir leur localisation et la forme de la cellule associée. Ainsi, ils s'occupent de la création, l'affectation et la suppression des données pour chaque modification de la structure de graphe. Deux types de graphes cellulaires coexistent : d'un côté les graphes cellulaires 2D et d'un autre les graphes cellulaires 3D. Les premier sont utilisés pour la saisie et les analyses 2D, les second sont utilisés pour la reconstruction 3D des méristèmes et pour les simulations.

Les deux classes de graphes sont pratiquement identiques, à ceci près que là où les graphes cellulaires 3D renvoient des positions en 3D, les graphes 2D renvoient des positions

en 2D. Sinon leurs interfaces sont identiques et elles dérivent toutes deux de la classe de graphe décrite dans la section suivante.

En plus des notions classiques d'un graphe, elles définissent la notion de cellules extérieurs (i.e. les ancres), de position des centroïdes et de polygone décrivant la géométrie d'une cellule.

Quand la saisie est réalisée à partir de parois, le graphe cellulaire 2D contient en plus un autre graphe qui relie les vertex. Dans ce graphe, les nœuds sont les vertex et les arcs sont les parois cellulaires. C'est ce graphe qui est construit par l'interface de saisie par les parois, le graphe des cellules étant calculé à partir du graphe des vertex (voir section II.3.2.2).

V.2.3.3 Autres structures de données

En plus des graphes et des graphes cellulaires, le noyau contient un ensemble d'algorithmes et de structures, dédiés principalement à la géométrie algorithmique et aux graphes. Ainsi, il intègre des algorithmes pour le calcul de surfaces, pour le remplissage de grille, pour la coloration de graphes...

V.2.4 Intégration d'un interpréteur

Pour pouvoir analyser et modifier les structures de données créées à partir des interfaces, nous avons choisi d'inclure un interpréteur interactif Python au programme (voir figure V.7). Cet interpréteur est relié à une fenêtre réalisant la boucle "Lecture-Évaluation-Affichage" (REPL pour "Read-Eval-Print Loop").

L'utilisation de Python dans un programme requiert un travail non négligeable de la part du développeur. Heureusement il existe des outils pour faciliter la création des modules d'extension Python, mais nous verrons quelques points particuliers à prendre en compte. Par contre, il n'existe pas d'outils permettant d'intégrer facilement un interpréteur interactif et de nombreux problèmes apparaissent, notamment pour l'interaction avec l'interface graphique.

Comme nous voulions garder le contrôle de l'interface graphique y compris pendant le fonctionnement de scripts python, nous avons choisi de lancer l'interpréteur dans un thread différent de celui de l'IHM. Il devient donc nécessaire de prévoir un mécanisme de communication entre le thread graphique et le thread Python pour permettre l'interaction entre les deux et aussi de prendre en compte le multi-threading lors de la création des modules Python.

V.2.4.1 Module d'extension de Python

L'interface Python/C++ consiste en une série de modules déclarant à Python les différentes structures de données auxquelles on désire accéder. Pour cela, nous avons utilisé Boost.Python, bibliothèque entièrement en C++ qui propose un ensemble de patrons* de classes et de fonctions pour déclarer les fonctions à exporter. Entre autres choses, Boost.Python prends en charge les problèmes de surcharge de fonction, de vérification de type, de traduction d'exception C++ en exceptions Python. Aussi, Boost.Python propose

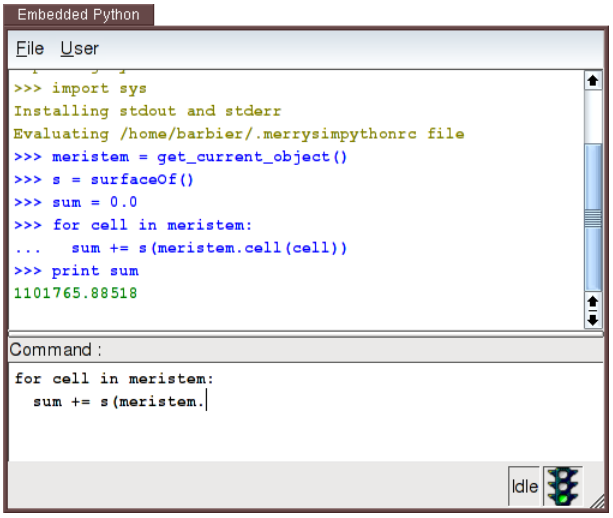


FIG. V.7: **Interface de l'interpréteur Python embarqué dans Merrysim.** L'interface est composée de deux fenêtres. Celle du dessus est la fenêtre d'historique. Les commandes utilisateurs apparaissent en bleu précédée de chevrons. La sortie standard du script Python est affichée en vert et la sortie d'erreur en rouge. Le fenêtre du bas est une zone d'édition dans laquelle l'utilisateur entre des instructions Python. Enfin, la barre d'état affiche une icône et un texte indiquant l'état de l'interpréteur et un menu configurable permet à l'utilisateur d'accéder rapidement aux commandes classiques.

des mécanismes permettant d'étendre les classes C++ en Python et d'utiliser des objets Python depuis le C++.

Pour permettre l'extension des objets C++ en Python, deux méthodes ont été utilisées. Dans le cas où l'objet C++ est l'instance d'un patron* (de classe ou de fonction), un objet spécial est créé qui encapsule un objet Python et appelle ses méthodes depuis le C++ en utilisant l'interface de Boost.Python. Dans le cas où l'objet C++ est prévu pour être dérivé, alors la classe encapsulant l'objet C++ pour Python utilise les mécanismes prévus par Boost.Python pour appeler les fonctions éventuellement réécrites en Python.

Interface Python de la bibliothèque de graphes. La bibliothèque de graphe étant au centre de l'application Merrysim, elle est exportée de façon a être entièrement utilisable depuis Python. De plus, le support du pickling (i.e. le système de persistance proposé en standard par Python) a été ajouté pour permettre le dialogue avec MGS (voir section V.3.1).

En ce qui concerne les possibilités d'extension de la bibliothèque en Python, ça n'est que partiellement fait pour l'instant. Ainsi, il n'est pas possible de créer une nouvelle classe d'arc entièrement en Python. Rendre la création possible ne serait pas très difficile (c'est prévu pour) mais relativement long et fastidieux car le nombre de méthodes est important et nous ne disposons pas d'outils permettant d'automatiser cette tâche.

Il y a tout de même un élément qui a été exporté en Python de façon à être redéfinissable, c'est la classe d'arc correspondant au graphe de Delaunay. Pour que ce soit faisable facilement, le paramétrage de la classe se fait par polymorphisme sur des classes utilitaires

plutôt que par instanciation de patron de classe. Les classes utilitaires ont été exportées en Python de façon à pouvoir en hériter et à utiliser les classes filles écrites en Python en lieu et place des classes C++.

Au final, il serait bien, dans un avenir proche, d'étendre l'interface Python de la bibliothèque de graphe de façon à pouvoir définir des classes d'arcs indifféremment en Python ou en C++ et pouvoir les utiliser depuis les deux langages indifféremment. Mais l'implémentation actuelle permet déjà d'utiliser intégralement, depuis Python, les classes existantes sans aucune limitation par rapport à une utilisation depuis le C++.

V.2.4.2 Interface Python/Qt

Comme nous l'avons vu dans la section précédente, l'interpréteur Python est exécuté dans un thread distinct de celui de l'application principale. C'est une méthode classique pour laisser l'accès à l'interface graphique alors que des calculs éventuellement longs sont en cours. Ainsi, du fait de la structure de l'interpréteur, toute commande Python doit être exécutée exclusivement dans le thread de l'interpréteur ou depuis un thread créé par l'interpréteur Python. Symétriquement, toute action graphique doit être exécutée exclusivement dans le thread principal de l'application : c'est une contrainte partagée par toute les bibliothèques graphiques actuelles.

Dans Merrysim, il est possible d'envoyer des commandes à l'interface graphique depuis l'interpréteur Python. Notamment, si l'utilisateur modifie une structure de donnée en cours d'édition depuis l'interpréteur, il doit informer l'interface de saisie qu'il est nécessaire de redessiner la structure pour prendre en compte les modifications. Nous avons donc un problème sur le lieu d'exécution de la requête, car la commande Python pour redessiner doit nécessairement être exécutée dans le thread Python, mais la commande d'affichage C++ doit, elle, être exécutée depuis le thread principal de l'application.

Pour résoudre ce problème, nous avons développé un système d'appel de fonction déporté pour la bibliothèque Qt afin de garantir l'exécution d'une fonction (ou méthode) depuis le thread principal de Qt. Par rapport à ce que propose la bibliothèque Qt, notre système ajoute la récupération de la valeur de retour et la transmission des exceptions qui ont été levées durant l'exécution de la commande. En pratique, les exceptions sont toutes interceptées et une exception est systématiquement levée dans le thread appelant, mais seules les exceptions dérivant de la classe `ThreadedException`, que nous avons définie, sont effectivement transmises (i.e. les autres lèvent une exception standard différente de l'exception originale).

V.2.4.3 Interface entre l'IHM et Python

Comme nous l'avons vu, l'IHM intègre une interface pour envoyer des commandes à l'interpréteur Python embarqué et remplacer les entrées/sorties standard des scripts.

Le système permettant d'envoyer des commandes à exécuter à l'interpréteur python est simplement une file combinée à un sémaphore dans un modèle producteur / consommateur. Une commande pouvant être soit une chaîne de caractère à exécuter, soit un objet python se comportant comme une fonction (i.e. contenant une méthode `__call__`) il est possible de créer un objet C++ avec en paramètre une fonction Python. Merrysim

propose une méthode pour garantir l'exécution de la fonction dans le thread principal de l'interpréteur Python.

Le remplacement des sorties standards et d'erreur se fait en créant des pipes* qui remplacent les descripteurs de sorties inclus dans le module `sys` de Python. Il aurait été possible de remplacer complètement les entrées/sorties standard du programme Merrysim mais nous n'avons pas retenu cette solution pour séparer les entrées/sorties Python des autres. Pour les entrées, Python utilise deux fonctions prédéfinies appelées `input` et `raw_input`. Nous les avons redéfinies et elle placent l'interface de l'interpréteur en attente d'entrées qui, au lieu d'être envoyées à l'interpréteur Python comme commande, sont alors renvoyées comme valeur de retour de ces fonctions.

V.2.5 Persistance des objets

La persistance des objets a été gérée différemment pour les objets créés par les interfaces de saisies et pour les résultats d'analyses ou les objets augmentés depuis Python.

Dans le premier cas, une classe implémentant le design pattern* de la factory* s'occupe de la sauvegarde et la lecture dans un format texte lisible de chacun des cinq types de fichier (méristème saisi par les parois, méristème saisi par les centres, cellules soeurs, position de la protéine PIN1, suivi dans le temps). Chaque fichier commence par une ligne de description du contenu puis contenant un numéro de version. Ainsi, il est possible de changer le format sans perdre les anciens fichiers.

Pour les objets créés (ou modifiés) depuis Python, nous avons choisi d'utiliser le système de persistance de Python : le pickling. Ce système repose sur une machine à pile spécialisée dans la construction d'objets. La sauvegarde consiste en l'écriture d'un programme qui, exécuté sur la machine de pickling, recréera tous les objets. C'est le système recommandé sur les listes de diffusion Python. Le principal avantage de ce système est de permettre la sauvegarde de tout objet Python pur et d'être supporté par Boost.Python (i.e. la bibliothèque propose des fonctions facilitant le pickling des objets C++ exportés en Python). Par contre, pour la suite, il serait souhaitable de rechercher une méthode plus complète car le pickling utilisé "tel quel" est trop dépendant de la structure du programme (par exemple, la structure des modules ne doit pas changer pour pouvoir relire un fichier) et il n'y a pas de gestion de version. Enfin, le langage choisi est tel qu'il est difficilement lisible par un humain sans être très facile à lire pour un programme (i.e. sans l'exécuter sur une vrai machine à pile). Il est donc très difficile d'intervenir sur ces fichiers une fois créés, ce qui pose des problèmes en cas de changement de structure du programme.

V.2.6 Choix des outils

V.2.6.1 Les langages

Le choix des langages utilisés pour le développement des logiciels Merrysim et Merryproj a été guidé par trois facteurs : rapidité de développement, efficacité à l'exécution et connaissance *a priori* du langage, et bien sûr l'adéquation dans le projet ALEA.

Lors de la réflexion sur le choix du langage à utiliser, nous avons distingué quatre usages nécessitant éventuellement des langages différents :

(i) le langage de script*
Nous avons besoin d'un langage à prise en main rapide, dynamique, qui ne nécessite pas de cycle explicite de compilation (i.e. interprété ou compilé à la volée) et pour lequel il existe un interpréteur (ou un JIT*). Il doit être facilement extensible, si possible dans différent langages, pour rendre accessible nos structures de données depuis ce langage. Enfin, il est plus que souhaitable que la communauté supportant le langage soit importante et active de façon à bénéficier d'une large palette d'outils.

(ii) l'architecture globale et l'interface graphique
Le problème principal est la stabilité et la facilité d'implémentation de l'architecture. Ainsi, la familiarité avec le langage utilisé devient très importante pour éviter les erreurs d'incompatibilité entre langage et architecture, car il est primordial que l'architecture soit fiable et raisonnable pour construire l'application.

(iii) les algorithmes coûteux en temps de calcul
Le premier critère est la vitesse d'exécution. Ainsi, on favorisera les langages compilés efficaces. Toutefois, nous chercherons tout de même un compromis entre vitesse d'exécution et familiarité avec le langage.

(iv) le moteur de simulation
L'objectif principal d'un langage pour la simulation est la facilité d'expression des hypothèses dans le langage. La rapidité peut toutefois devenir un critère important en fonction des modèles implémentés.

(i) Le langage de script* étant au cœur de la plate-forme ALEA, la réflexion a été menée au niveau du projet et non de la thèse. En plus des critères choisis pour le projet, il devait être facile d'accès pour des biologistes. Le langage Python (van Rossum, 2005) répondant correctement à ces critères, il a été choisi pour le langage de script. Les points forts reconnus de Python sont la facilité d'apprentissage, sa très large communauté et le grand nombre de bibliothèques disponibles.

(ii) Lors du choix de l'architecture globale de Merrysim, l'argument déterminant a été ma familiarité avec le C++ et mes expériences passées dans l'architecture d'applications écrites en C++. C'est donc le langage qui a été choisi pour l'architecture. Quant à l'interface graphique, l'existence de la bibliothèque Qt Trolltech, écrite en C++ et reconnue comme une des meilleurs bibliothèques graphiques, renforce ce choix.

(iii) Pour le développement des algorithmes coûteux en temps de calcul (par exemple pour le traitement d'image), du fait de l'architecture objet du logiciel, de la facilité à utiliser les très nombreuses librairies écrites en C et pour ne pas multiplier inutilement les langages, il a été décidé d'utiliser le C++.

(iv) Enfin, pour le moteur de simulation, un objectif supplémentaire était d'évaluer l'utilisation d'un langage dédié à la modélisation des systèmes dynamiques à structure dynamique. Le choix s'est porté vers MGS (voir section IV.2 ; Giavitto et Michel, 2001b) qui est un langage fonctionnel déclaratif. Déjà avec des structures plus simples, des langages déclaratifs tels que les L-systèmes ont démontré leur intérêt, car les modèles sont le plus souvent exprimés par des déclarations locales du comportement. Aussi l'existence de collaborations antérieures entre l'équipe développant le langage et notre équipe a probablement facilité les interactions.

V.2.6.2 Les bibliothèques

Les bibliothèques utilisées ont été choisie principalement sur trois critères : la qualité de l'implémentation et de l'interface, la facilité d'utilisation et la communauté autour de la bibliothèque. Par ailleurs, toutes les bibliothèques utilisées devaient être compatible avec une licence de distribution libre pour Merrysim.

Nous avions besoin de bibliothèques principalement pour :

1. l'interface graphique
2. l'interface Python/C++
3. les graphes
4. le calcul scientifique
5. le calcul statistique
6. la manipulation d'images

L'interface graphique. Pour l'interface graphique, les choix envisagés ont été : GTK+ (website GTK+), Qt (Trolltech) et wxWidgets (website wxWidget; Smart *et al.*, 2005). Ces bibliothèques sont les trois bibliothèques compatibles avec une licence libre*, avec de nombreux utilisateurs, multi-plate-forme (i.e. elles existent pour Linux/X11, Windows et MacOS X) et disponibles à la fois en C++ et en Python.

GTK+ est une bibliothèque écrite en C qui est réputée pour la qualité de sa conception. Elle a été créée en 1996 comme remplacement à Motif pour le logiciel de dessin "the GIMP" (website The Gimp). Mais c'est maintenant une bibliothèque très largement utilisée, notamment par le projet GNOME (website GNOME). Il existe des interfaces à GTK+ pour de nombreux langages, et notamment pour le C++ avec GTKmm (website Gtkmm) et Python avec PyGTK (website PyGTK). GTK+ dispose d'un créateur d'interface graphique appelé Glade qui produit des fichiers XML indépendant du langage qui sera utilisé pour l'application.

Qt est une bibliothèque écrite en C++ elle aussi réputée pour la qualité de sa conception. Elle a été créée par Trolltech, une entreprise allemande, qui prend en charge son évolution. Qt est distribué avec une double licence : avec la licence propriétaire, payante, l'application utilisant Qt peut être distribuée sans aucune condition, mais Qt est aussi distribuée sous licence GPL, ce qui impose que toute application se basant sur cette version de Qt soit distribuée elle aussi sous licence GPL. Cette bibliothèque est très utilisée dans la communauté libre : elle est notamment à la base du projet KDE (website KDE). Il existe une interface pour le langage Python avec PyQT (Rempt, 2001). Comme GTK+, Qt dispose d'un créateur d'interface graphique appelé Qt-Designer qui produit aussi des fichiers XML indépendant du langage de l'application. Toutefois, Qt-Designer est clairement orienté vers la création d'applications en C++ avec la possibilité d'associer un fichier C++ au fichier XML et de l'éditer depuis Qt-Designer.

wxWidgets (ex-wxWindows) est une bibliothèque écrite en C++ distribuée sous une licence open source (i.e. approuvée par l'Open Source Initiative (website OSI)) compatible GPL. Bien que relativement ancienne (elle a été créée en 1992), elle n'est devenue importante que très récemment avec l'essort de C++ dans le monde du libre et la nécessité d'avoir une bibliothèque graphique C++ libre sur toutes les plate-forme (Qt sous Windows n'est distribuée sous licence GPL qu'à partir de la version 4 dont la première pré-version n'est sortie qu'en juin 2005). L'API de wxWidgets est largement inspirée de la partie interface graphique des Microsoft Foundation Classes (MFC), l'objectif étant de facilité la transition des développeurs ayant l'habitude des produits Microsoft. Contrairement aux deux bibliothèques précédentes, wxWidgets a été développée au-dessus d'autres bibliothèques graphiques (originellement les MFC sous Windows et XView sous UNIX/X11) aussi il existe maintenant un grand nombre d'implémentations pour des couples bibliothèque graphique / système d'exploitation. Le port pour Python est wxPython (website wxPython), mais il n'est disponible que pour la version GTK de wxWidgets. wxPython ne dispose pas de créateur d'interface graphique officiel mais plusieurs ont été développés, soit pour une seul langage (wxWorkshop, Boa), soit pour plusieurs (wxDesigner).

Choix de la bibliothèque graphique. wxWidgets a rapidement été écarté car sa syntaxe proche des MFC est loin d'être aussi souple de celle proposée par Qt ou GTK+. De même, leur choix de reposer sur une autre bibliothèque graphique est, à mon avis, une erreur, déjà commise par Sun lors de la création de Java avec AWT (Sun a d'ailleurs changé d'optique avec l'introduction de Swing dès Java 1.2) car il devient laborieux de créer une interface agréable et efficace sur toutes les plate-formes, les différentes bibliothèques pouvant avoir des comportements très différents sur certains objets (ces différences sont le plus visible sur les menus déroulant, la gestion du focus*, ...).

Entre Qt et GTK+ c'est à la fois la qualité de l'interface Python et l'historique de l'équipe qui nous ont fait choisir Qt. Au moment des tests, PyGTK avait tendance à provoquer des erreurs de segmentation de l'interpréteur Python quand il aurait du lever une exception Python comme savait déjà très bien le faire PyQT.

Interface Python/C++. Pour interfacer Python et C++ nous avons isolé quatre possibilités :

1. l'API* C native de Python ;
2. SIP (website SIP), qui a été développée pour PyQT ;
3. Swig (website Swig), une bibliothèque multi-langage ;
4. Boost.Python (Abrahams, 2003), une bibliothèque qui fait partie du projet Boost et qui est spécialisée dans la création d'interfaces de Python/C++.

La première solution a rapidement été écartée car l'API* native de Python est bien trop complexe pour être utilisée pour la création d'interface avec un grand nombre de classes. Quant à SIP, au moment de l'étude la documentation était inexistante et il n'a donc pas pu être évalué.

Le choix a alors du se faire entre Boost.Python et Swig. Les deux bibliothèques sont reconnues comme étant bien conçues.

Swig vise à être une bibliothèque générale d'extension en supportant des bibliothèques en C et C++ mais surtout en permettant la création d'extensions pour beaucoup de langages (Python, Ruby, Java, OCaml, ...). L'export se fait par un ensemble de directives qui peuvent être incluses dans les commentaires des entêtes⋆. Un pré-processeur⋆ va alors générer le code nécessaire pour la bibliothèque d'extension. Les classes exportées seront toutefois toutes enveloppées dans des classes natives Python, ce qui permet une très bonne intégration dans le langage, mais réduit les performances des appels.

De son côté l'équipe développant Boost.Python a choisi de ne cibler que l'extension de Python avec C++ et d'utiliser pour générer le code d'extension les patrons C++. De plus, Boost.Python offre une interface C++ aux objets Python qui facilite la création de bibliothèques C++ manipulant des objets Python. L'utilisation massive de patrons pose toutefois des problèmes du fait des compilateurs : les différents compilateurs ne se comportent pas de la même façon et il existe de nombreux bugs, et en ce qui concerne G++ une telle utilisation de patrons entraîne une utilisant excessive de mémoire, ce qui impose un découpage artificiel pour la compilation des extensions. Enfin, la taille des extensions générée par Boost.Python sont jusqu'à 10 fois plus grosses que celles générées par Swig.

Au final, nous avons retenu Boost.Python pour son intégration dans C++ et parce que Swig ne supportait pas suffisamment bien le C++ au moment du choix. Notamment, l'existence de fonctions ou méthodes surchargées rendait impossible l'utilisation de Swig.

Les graphes. En recherchant les bibliothèques de graphe pour le C++, nous en avons trouvé seulement une seule de libre : la Boost Graph Library (Siek *et al.*, 2001) du projet Boost. Cette bibliothèque propose une série de concepts⋆ adaptés aux graphes. Elle propose aussi un ensemble d'algorithmes reposant sur ces concepts ainsi que deux implémentation de graphes respectant chacune des concepts différents. Même si la hiérarchie de concepts et les algorithmes se sont révélés très intéressant et bien construits, les implémentations de graphes se sont révélées difficiles à utiliser avec des erreurs complexes à comprendre et des problèmes de conception important (comme par exemple l'impossibilité de supprimer les données associées à un nœud lors de sa suppression).

Aussi, nous avons décidé d'implémenter notre propre bibliothèque de graphe en suivant une API compatible avec celle définit dans le cadre du projet ALEA et en ajoutant les fonctions nécessaires pour respecter les concepts de la BGL que notre implémentation pouvait respecter.

Le calcul scientifique. Pour le calcul scientifique, il existe deux bibliothèques majeures pour Python : Numarray (Greenfield *et al.*, 2003) et SciPy (website SciPy). Numarray est une bibliothèque de calcul matriciel et SciPy est un ensemble de bibliothèques dédiées au calcul scientifique. Bien que SciPy propose aussi une bibliothèque de calcul matriciel (Numeric) proposant des fonctionnalités proches de Numarray, nous avons utilisé préférentiellement Numarray quand SciPy n'était pas utile car il est plus souple d'utilisation et plus complet.

Le calcul statistique. SciPy propose quelques fonctions statistiques simples. Toutefois, pour des calculs plus complexes (e.g. estimation de densité de probabilité) nous avons

utilisé RPy (website RPY) qui est une interface Python au logiciel de statistique GNU R (website R).

La manipulation d'images. Pour la manipulation d'images, la bibliothèque de référence en Python est la "Python Image Library" (website PIL). C'est donc elle que nous avons utilisé lorsque les manipulations d'images sont faites en Python. Pour les manipulations faites en C++, nous avons utilisé soit les objets et filtres fournis par la bibliothèque Qt, soit nos propres objets et filtres quand Qt ne proposait pas ce dont nous avions besoin (notamment pour les filtres topologiques).

V.3 Moteur de simulation

Les simulateurs ont été écrits en utilisant le langage MGS. C'est un langage interprété expérimental dédié à la simulation. Il est développé en OCaml et fonctionne dans un processus distinct de l'application principale. Nous avons donc écrit un système de communication pour pouvoir utiliser, depuis Merrysim, les scripts écrits pour MGS.

V.3.1 Interface Python/MGS

Un des points clefs de l'intégration d'un moteur de simulation utilisant MGS dans Merrysim réside dans la communication et dans l'API de dialogue proposée. MGS fonctionnant nécessairement dans un processus distinct, il est nécessaire d'utiliser un médium de communication inter-processus.

L'interface Python/MGS fonctionne à deux niveau : le premier niveau permet d'interagir depuis Python avec MGS en envoyant des commandes, des objets, en récupérant la sortie, les erreurs et les objets envoyés par le script MGS. Le second niveau offre une abstraction pour la simulation avec des notions d'objet initial, d'initialisation, de transformation...

Premier niveau. Étant donné sa nature expérimentale, MGS ne dispose que de très peu de fonctions d'entrée/sortie et toutes sont liées à la notion de fichier. C'est toutefois largement suffisant, car les plate-formes UNIX proposent une API dans laquelle "tout est fichier". Ainsi, les fichiers servent non seulement au stockage des données, mais aussi au dialogue avec les différents périphériques ou entre les processus. Le système de dialogue inter-processus par fichier est appelé système par pipes*, et si ces pipes sont instanciés dans l'arborescence du système de fichier, on parle de "pipe nommé". MGS a alors besoin uniquement du chemin (i.e. le "nom") vers les pipes nommés. Un des intérêts de ce système est qu'il est facile de remplacer un des pipes nommés par un fichier. Ni MGS ni Merrysim ne verra la différence et il est facile de découpler le débuggage des deux parties. Étant lancé par le script Python, les arguments du script MGS contiennent les noms des trois pipes utilisés : un pour la lecture, un pour l'écriture et un pour les rapports d'erreurs. En même temps, le script lance deux threads d'écoute (pour la sortie et pour les erreurs).

Le protocole de communication entre MGS et Python définit un ensemble de commandes asynchrones qui n'attendent jamais de réponse. Ainsi, chaque partie doit prendre l'initiative d'envoyer des données à l'autre partie : le dialogue est symétrique, il n'y a pas

vraiment de relation maître/esclave. La seule exception est l'existence d'un commande pour que Python puisse faire se quitter MGS et pas l'inverse.

Le script de lancement de MGS remplace l'IDLE standard par une fonction qui se met en écoute sur le pipe* de lecture. L'IDLE attends de lire une ligne complète correspondant à une des 6 commandes reconnues :

- `object` attends ensuite un nom et une expression : affecte le résultat de l'expression au nom ;
- `command` attends une commande MGS et l'exécute ;
- `script` attends le nom d'un script et le charge avec la fonction `use_module` ;
- `echo` attends une chaîne de caractère et l'écrit sur le pipe de sortie ;
- `listvar` renvoie la liste des variables actuellement définies ;
- `stop` quitte l'interpréteur.

En même temps que l'IDLE, un certain nombre de fonctions sont fournies pour permettre de générer le pickling d'un objet. Ainsi, l'envoie d'un objet à Python se fait par l'intermédiaire de la fonction `write_object` qui prends deux arguments : la fonction de pickle et l'objet. La fonction de pickle est soit une des fonction standard si l'objet à envoyer est simple (i.e. transformable en un des objets Python de base : liste, dictionnaire, nombre, chaîne de caractères) soit une fonction définie par l'utilisateur en utilisant les fonctions fournies pour le pickling vers des objets Python.

Du côté de Python, un thread est lancé pour intercepter les objets envoyés par MGS. Ce thread attends dans un premier temps la taille des données à recevoir, puis l'objet sous forme de pickle. Si l'unpickling réussi, l'objet obtenu est envoyé au premier gestionnaire enregistré s'il y en a un. Tout d'abord, la fonction `can_handle` est appelée pour déterminer si le gestionnaire prend en charge l'objet. Si oui, alors le module `MGS` envoie l'objet à la fonction de traitement du gestionnaire (i.e. `__call__`) et récupère le nouvel objet en sortie. Tous les gestionnaires sont successivement appelés dans l'ordre définit par l'utilisateur, à moins que l'un d'eux renvoie `None`, auquel cas la procédure s'arrête.

Pour envoyer un objet vers MGS, il est nécessaire que l'objet respecte l'interface `MgsSendInterface`. Cette interface définit deux méthodes : une méthode `get_type` qui renvoie le type de l'objet envoyé (i.e. une des 6 commandes reconnues par l'IDLE) et une méthode `marshall` qui renvoie une chaîne correspondant à la commande. Si le type est `object`, alors il faut en plus que l'objet ait un attribut `name` qui soit le nom MGS à attribuer à l'objet, ce qui est requis par l'interface `MgsSendObjectInterface`. Le nombre de lignes de la commande finales est calculées par le module et envoyé avant la commande elle-même.

Du côté Python, chaque commande correspond à l'envoie d'un objet instance d'une classe portant le nom de la commande, sauf pour la commande `object` qui correspond à l'envoie de l'objet directement.

Pour la communication de MGS vers Python, MGS définis un certain nombre de fonctions qui envoie des objets Python sous forme de pickling. Du côté de Python, des gestionnaires qu'il faut enregistrer auprès de l'objet gérant le protocole de communication définissent les actions à entreprendre en fonction des objets reçus. Il existe des gestionnaires prédéfinis, comme `StoreHandler` qui conserve les objets d'un certain type ou `DiskStoreHandler` qui joue le même rôle mais les sauvegarde sur disque au lieu de les

conserver en mémoire. Mais il est virtuellement possible d'envoyer n'importe quel objet de MGS vers Python et il est donc possible de définir un protocole de plus haut niveau.

Second niveau Le second niveau est un exemple de protocole de plus haut niveau, avec ses propres signaux de contrôle, définit sur le premier niveau.

Son but est de permettre de raisonner en terme de modèle et de simulation. Ainsi, il permet de définir la fonction de transformation, de transformation initiale, de transformation "cosmétique" (i.e. qui n'est pas prise en compte pour les calculs mais qui est appliquée juste avant d'envoyer l'objet à Python), le nombre de sorties voulues et le nombre de pas de calcul entre deux sorties. Il ajoute aussi un signal informant l'utilisateur de la fin de la simulation. Il comporte trois classes : la première est l'interface utilisateur et s'appelle `Model`, la seconde permet de spécifier les attributs non standards à récupérer depuis MGS, la troisième s'occupe de l'ordonnancement des commandes (MGS ou Python).

La classe `Model` définit des méthodes pour lancer la simulation depuis un script ou graphiquement (uniquement si l'interface graphique de Merrysim est lancée). Elle définit aussi la liste des scripts MGS contenant le modèle pour la simulation.

Les attributs non standards (i.e. autres que ceux stockés dans le graphe de cellule), nécessitent de spécifier la fonction MGS qui sera utilisée pour les convertir en objet Python. Par défaut, une fonction décidera d'un comportement standard à partir du type MGS (i.e. les nombres sont envoyés tels quels, les monoïdes sont transformés en listes...). Mais il est possible de demander explicitement de n'envoyer que les booléens ou des nombres à virgule flottante par exemple.

Enfin, l'ordonnanceur se comporte comme un système de batch : il exécute les commandes une par une, dans l'ordre où elles sont spécifiées. L'accès à cet ordonnanceur se fait soit par la méthode `post` de la classe `Model` qui permet d'envoyer une nouvelle commande à l'ordonnanceur, soit avec la propriété `scheduler` de la même classe pour un accès direct à l'objet. L'accès direct permet, outre l'envoie de commandes, de voir quels sont les objets qui ne sont pas encore traités, et éventuellement d'agir sur l'ordre des commandes. Une évolution intéressante serait de structurer la liste des commandes de façon à déplacer des groupes logiques (par exemple toutes les commandes correspondant à une même étape de simulation voire à une même simulation).

V.4 Merryproj

Outre la nécessité d'avoir un logiciel permettant de reconstruire la vue de dessus d'un méristème à partir d'une acquisition au microscope confocal, ce logiciel a aussi permis de tester l'usage de la boite à outil graphique PyQT (Rempt, 2001) pour le développement d'un petit programme.

Ce programme est très simple comparé à Merrysim. Il possède un but unique : sélectionner une série d'images correspondant à une acquisition d'un méristème apical caulinaire à l'aide un microscope confocal (ou tout autre microscope à fluorescence permettant d'obtenir une série de coupes optiques, comme les microscopes multi-photoniques (Xu *et al.*, 1996)) et calculer la vue de dessus du méristème.

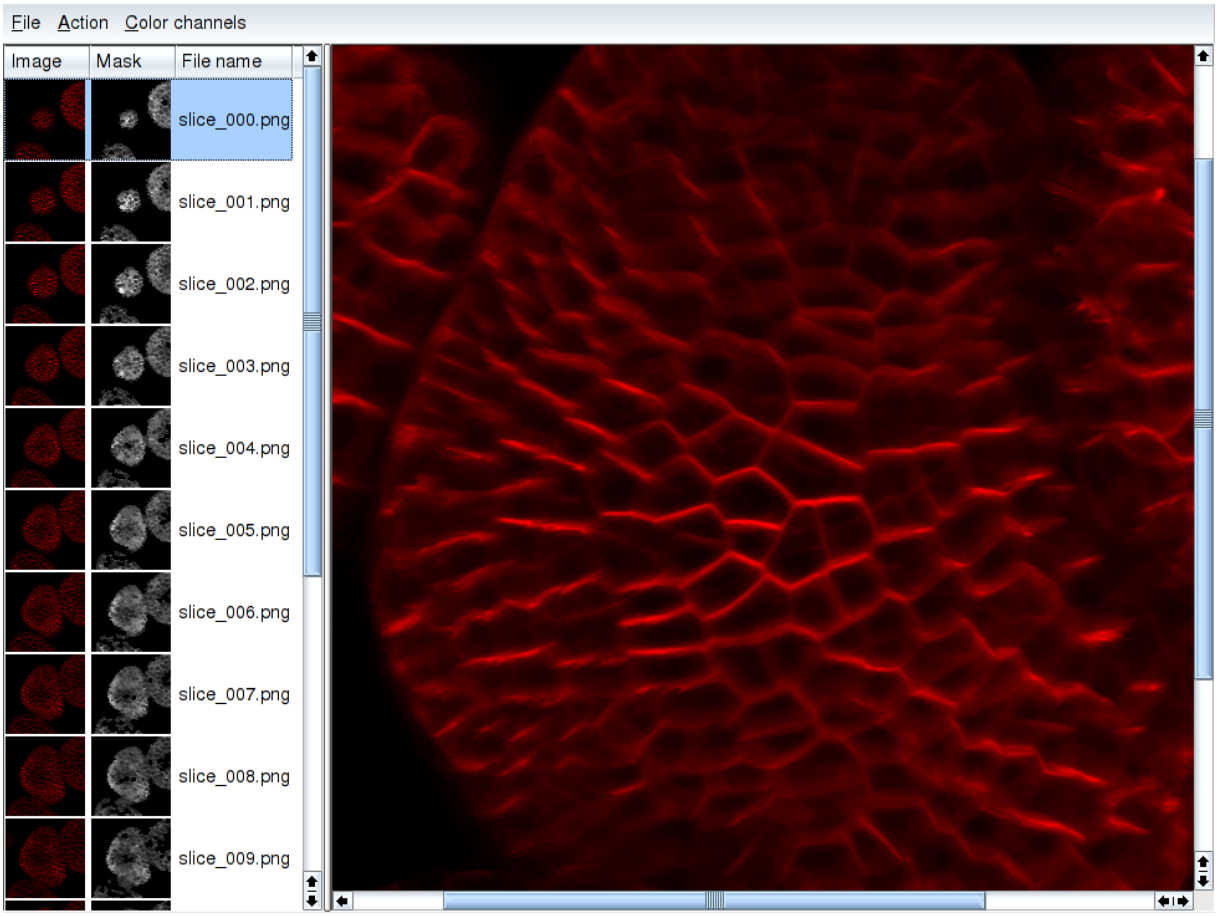


FIG. V.8: **Interface principale du logiciel Merryproj.** Le panneau de gauche montre des images miniatures des coupes optiques et du masque s'il a été calculé. Le panneau de droite montre ici le résultat du calcul : donc la projection associée à la coupe la plus haute.

V.4.1 Structure générale.

Le programme est structuré autour d'une classe principale correspondant à la fois à l'application et à sa fenêtre principale. La fenêtre comporte une barre de menu qui permet de lancer les différentes actions et de deux panneaux pour afficher la liste des coupes optiques à gauche et la coupe (ou la projection) sélectionnée à droite (voir Figure V.8), chaque panneau étant géré par une classe dédiée.

La création d'une nouvelle projection commence par la sélection et l'ordonnancement des images correspondant aux coupes optiques. Cette opération est réalisée à l'aide d'une interface dédiée (voir Figure V.9) conçue à l'aide du créateur d'interface de Qt (Trolltech) : `qt-designer`. Qt-designer est prévu à l'origine pour travailler avec la version C++ de Qt et offre donc la possibilité d'associer du code C++ à la boite de dialogue. Toutefois, PyQt contient un programme appelé `pyuic` qui permet de générer le code Python correspondant à la boite de dialogue. La partie fonctionnelle de la boite de dialogue se fait alors de manière

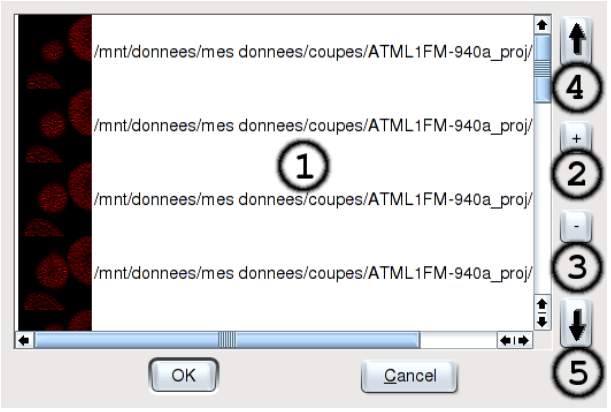


FIG. V.9: **Ouverture d'une nouvelle projection.** (1) La liste des coupes optiques ordonnées avec la coupe du dessus en haut.
(2) et (3) sont les boutons permettant d'ajouter/supprimer des images.
(4) et (5) permettent de réordonner les images en déplaçant une image vers le haut/le bas.

classique : en créant une classe dérivée de celle générée par `pyuic` et en implémentant les `slots` définis lors de la création de la boite de dialogue.

Les calculs sont coordonnés par une classe Python contenant une méthode permettant de calculer le masque de transparence pour une coupe optique et de l'appliquer à une image de fond. Cette même classe permet de calculer les images réduites pour le panneau de gauche. Toutefois, l'implémentation effective des méthodes de calcul (i.e. les bassins versants et la fermeture topologique, voir chapitre II) sont implémentés dans une bibliothèque externe écrite en C++. L'interface entre la bibliothèque C++ et Python a été faite en utilisant la bibliothèque Boost.Python (Abrahams, 2003).

V.4.2 Structure de la bibliothèque C++.

La bibliothèque C++ comporte trois classes correspondant à des tableaux d'entier 1D, 2D et 3D dont la taille est fixée à la construction. Le tableau 2D comporte principalement une méthode pour calculer les bassins versant sur l'image correspondante au tableau ainsi qu'un des méthodes pour tronquer ou encore optimiser la plage des valeurs utilisées. Le tableau 3D comporte lui simplement les méthodes nécessaires pour calculer la fermeture topologique sur des valeurs binaires et pour "projeter" le résultat d'une fermeture dans un tableau 2D.

La structure de ces classes a été dictée avant tout par un soucis d'efficacité en essayant de fournir un minimum de facilité d'utilisation. Ainsi, pour limiter le nombre de déréférencement ou de calcul nécessaires pour accéder à une valeur, les grilles 2D et 3D sont implémentées sur une structure linéaire. Ainsi, dès que possible le parcours est fait systématiquement, incrémentant des pointeurs et des valeurs plutôt que multipliant des indices ou déréférançant des pointeurs (ce qui est fait implicitement pour des tableaux multi-dimensionnels en C ou C++).

Toutefois, cette optimisation n'est pas utilisée pour l'extension Python de ces classes car c'est une optimisation inutile devant le temps utilisé pour un simple appel d'une

fonction C++ depuis Python et la conception d'algorithmes en suivant cette contrainte est plus complexe.

V.5 Discussion

V.5.1 Langage principal pour l'architecture de la plate-forme

Une des questions importante est de déterminé quel langage il vaut mieux utiliser pour implémenter l'architecture globale de la plate-forme ALEA. Comme Merrysim a été développé très tôt, avant que le choix de Python ne soit pris pour le projet ALEA, l'architecture principale a été réalisée en C++. Intégrer un interpréteur Python à une application C++ s'est révélée une tâche difficile et pleine de subtilités. Le GIL de Python ne pouvant pas être testé, il a fallu faire très attention à ne pas provoquer de deadlock* avec les locks de l'application. Par ailleurs, l'interaction avec Qt a été elle-aussi difficile à mettre au point du fait de ce même GIL et de la nécessité de manipuler les différents verrous et sémaphores correctement. Enfin, les fonctions d'exécution de code Python depuis un programme en C sont plus complexe que leurs équivalent Python et nécessitent une analyse des instructions envoyées.

Tous ces problèmes de synchronisation, même s'ils existent, sont bien moins présent dans le cadre d'une application écrite en Python entièrement. Car alors le GIL n'est plus à prendre en charge et il ne reste plus qu'à garantir des opérations atomiques sur les structures de données du programme. De plus, une application Python pourra facilement bénéficier des modules écrits en Python pour l'analyse de code Python. Ainsi, il devient plus facile de proposer la complétion des termes, la présentation de la signature de la fonction... comme le propose l'application PyAlaMode par exemple.

Aussi, le fait d'utiliser Python pour l'interface graphique et l'architecture principale simplifie grandement le dialogue entre l'interpréteur embarqué et le thread Qt.

Ainsi, nous recommandons l'usage de Python pour l'architecture principale du programme dans le cas de l'intégration d'un interpréteur car la gestion des threads Python depuis le C ou le C++ est une tâche fastidieuse et difficile.

V.5.2 Techniques de programmation générique C++ et extensions Python

Écrire du code générique C++ peut être fait principalement par deux moyens : le polymorphisme ou les patrons. Nous allons voir les avantages et les inconvénient de chacune de ces deux méthodes.

Généricité par polymorphisme*. La méthode la plus ancienne consiste à utiliser la dérivation et une approche purement objet. Les algorithmes sont implémentés sur une classe de base que l'utilisateur devra dériver pour spécifier les parties paramétrables.

L'avantage de cette méthode est que la résolution de la généricité se fait à l'exécution et objet par objet. Ainsi, une liste implémentée par cette technique pourra stocker des objets de types dynamiques différents pourvu que leur classe dérive d'un même classe de base

définie lors de la création de la classe de liste. Par contre, cette résolution à l'exécution a un coût en temps d'exécution et en place mémoire des objets.

Du point de vue de la construction d'extensions Python, cette solution a un très gros avantage : la plupart des bibliothèques proposent des systèmes pour pouvoir étendre la classe de base depuis Python car c'est une technique de programmation générique commune à C++ et Python.

Généricité par patrons. Une technique beaucoup plus récente pour la généricité est l'usage des patrons* de classe ou de fonction. C'est une technique qui a d'abord été popularisée par la STL[1] puis qui a été plus récemment utilisée intensivement par l'ensemble de bibliothèques Boost (website Boost).

Le principal avantage de cette méthode est que la résolution de la généricité se fait à la compilation. Ainsi il n'y a aucune perte de temps du à cette généricité lors de l'exécution. Aussi, la généricité proposée par cette méthode est bien plus poussée et les possibilités de génération de code conditionnels permettent de créer des bibliothèques à l'utilisation très simple. Par contre, les temps de compilation et la taille de code généré peuvent facilement être très importants du fait des patrons. Par ailleurs, l'édition des liens est rendue plus difficile et il y a des précautions particulière à prendre, notamment sous MS Windows[2].

Du point de vue de la construction d'extensions Python, cette approche a un défaut majeur : il n'existe pas d'équivalent Python à cette structure et il n'est même pas possible d'exporter un patron* en Python. Une solution consiste à exporter divers instances du patron. S'il y en a beaucoup, il y a d'abord un problème de nommage en Python du patron (surtout s'il s'agit de classes car pour les fonctions, Boost.Python peut éventuellement s'occuper de gérer la surcharge). Ensuite, écrire le code nécessaire pour exporter les instances peut devenir fastidieux. Il devient alors utile d'écrire un patron de la fonction d'export Python (ou de classe) permettant d'exporter les patrons instanciés. Toutefois, une bonne définition du patron de la fonction d'export qui marche quel que soit l'instance peut être assez complexe à réaliser. La tâche a été entreprise pour les types définis par la STL, mais non seulement il reste des restrictions, mais il a fallu plusieurs semaines pour mettre au point les patrons de classe et de fonction utilisées dans ce cas. Enfin, s'il est en plus nécessaire de rendre le patron extensible depuis Python, il y a alors deux possibilités. La première consiste à instancier le patron pour un objet Python, et il est recommandé d'utiliser la classe `boost::python::object` de la bibliothèque Boost.Python plutôt que le `PyObject*` proposé par l'API de Python. L'inconvénient de cette méthode est qu'elle nécessite probablement de réimplémenter le patron pour cet objet. Une autre technique consiste à créer un proxy C++ encapsulant l'objet Python. Ainsi, tout le code de conversion Python vers C++ et vice-versa est restreint à cette classe et il est très probablement possible d'utiliser l'implémentation générique du patron de classe ou de fonction.

[1]Standard Template Library, bibliothèque standard en C++ qui propose un ensemble de structures de données et des fonction d'entrée/sortie de haut niveau

[2]voir `http://wiki.python.org/moin/boost.python/CrossExtensionModuleDependencies` pour discussion

Cas d'utilisation de ces méthodes. La décision d'utiliser l'une ou l'autre des méthodes dépendra principalement de deux paramètres : le premier concerne le type de généricité voulue, le second est le langage-cible principal.

Tout d'abord, s'il est nécessaire que la généricité soit résolue à l'exécution, alors seul le polymorphisme* permettra de résoudre le problème. Par contre, s'il est nécessaire de générer un code conditionnellement à des arguments, alors seul les patrons* pourront être utilisés. Il faut toutefois noter qu'il est possible de mélanger patron et polymorphisme si la généricité est résolue tantôt à l'exécution, tantôt à la compilation.

Dans les nombreux cas où aucune de ces conditions n'est vérifiée, il est possible d'utiliser les patrons avec parcimonie, par exemple en évitant les objets à exporter en Python. De même, réduire le nombre de paramètre des patrons permettra de réduire le nombre d'instances à exporter en Python.

Sinon, il est utile de considérer la cible principale de tel ou tel algorithme ou objet. Ainsi, un algorithme écrit pour être compatible avec la Boost Graph Library sera nécessairement écrit en template. Par contre, il sera peut-être suffisant de l'exporter pour la classe de graphe implémentée dans Merrysim (ou une autre) pour pouvoir l'utiliser depuis Python. Inversement, si un algorithme est destiné a être quasi-exclusivement utilisé depuis Python, il est pertinent d'éviter les templates et de préférer le polymorphisme*. Dans ce dernier cas, le surcoût lié à l'utilisation de Python rend de toute façon négligeable la perte due au polymorphisme. Dans le cas d'un usage équilibré des deux côtés, il devient pertinent d'utiliser les patrons* mais de n'exporter qu'une instance utilisant une classe prévue pour être dérivée en Python.

V.5.3 Une interface multi-public

Une des problématique récurrente dans les recherches à l'interface de plusieurs domaines est d'arriver à ce que tous les chercheurs profitent des outils développés. Ainsi, notre problème est de mettre à disposition de tout le monde, informaticiens et biologistes, les outils que nous avons développés. Ainsi, nous avons choisi de présenter une interface double : graphique et langagière. L'interface graphique permet d'utiliser l'application plus rapidement, l'apprentissage étant plus facile, mais les possibilités d'actions sont plus restreintes que pour l'interface langagière. L'interface langagière est elle plus complexe à maîtriser, mais elle permet d'agir beaucoup plus librement que l'interface graphique. Aussi, pour permettre à un maximum de personnes d'utiliser l'interface dans toute sa souplesse, le langage choisi pour l'interface l'a été pour sa facilité d'apprentissage, au moins pour les structures de contrôle simples.

Par rapport aux applications réalisées, Merryproj et Merrysim sont aujourd'hui utilisés de manière autonome par les biologistes du laboratoire de biologie cellulaire de Versailles. L'utilisation de Merrysim peut aller de la saisie jusqu'à la simulation. Pour le moment, seules les méthodes d'analyses ne sont pas accessible sans formation relativement poussée à la plate-forme et au langage Python.

Conclusion

Résumé des travaux

L'objectif de cette thèse était le développement et l'utilisation d'outils et méthodes mathématiques et informatiques pour l'étude du méristème apical caulinaire. Nous avons décliné cet objectif selon trois aspects. Du point du vue biologique, nous avions l'ambition d'avancer dans la compréhension du fonctionnement du méristème apical caulinaire et aussi de déterminer les zones d'ombres où de nouvelles recherches seraient pertinentes. Pour cela, nous voulions développer les outils mathématiques et informatiques utiles à l'étude du méristème, du point de vue morphologique, physiologique ou génétique. Enfin du point de vue génie logiciel, nous voulions déterminer l'efficacité de méthodes et de techniques dans le cadre du développement d'outils destinés à la biologie du développement.

Nous avons montré au chapitre II qu'il était possible d'automatiser un certain nombre de tâches de digitalisation répétitives. Notamment, en utilisant les propriétés géométriques du méristème, il est possible d'extraire sa surface des images obtenues par microscopie confocale. Ensuite, nous avons décrit une heuristique permettant de retrouver la lignation cellulaire à travers un ensemble de reconstructions géométriques d'un même méristème dans le temps, en prenant en compte les divisions cellulaires. Tout ceci a été rendu possible en développant et en utilisant des outils mathématiques et informatiques et en les adaptant à l'objet étudié : le méristème.

Dans le chapitre III, nous avons vu comment les techniques issues de l'informatique et de la biologie peuvent être utilisées de concert pour l'étude de la physiologie du méristème. La formalisation des hypothèses biologiques a permis la simulation de flux sur des méristèmes digitalisés. De ces simulations a émergé un résultat inattendu : il y a de l'auxine dans le centre. À la fois en testant divers jeux d'hypothèses et de paramètres grâce au moteur de simulation créé pour l'occasion et en menant d'autres expériences génétiques, pharmacologiques ou physiologiques nous avons obtenu un faisceau convergent de données confirmant cette première prédiction.

Dans le chapitre IV, nous avons étendu le modèle utilisé au chapitre précédent pour y inclure la croissance du tissus méristématique. Nous avons aussi ajouté les hypothèses manquantes, notamment sur le positionnement de la protéine PIN1. Un premier modèle dynamique a été développé. Il nous a permis d'évaluer l'utilisation du langage MGS pour notre problème de modélisation, et nous avons montré la faisabilité d'un modèle discret intégrant une croissance individualisée de cellules, la division cellulaire, et la régulation de l'état physiologique des cellules par un système d'équations différentielles, soit un modèle entrant dans la catégorie des systèmes dynamiques à structure dynamique.

Enfin, dans le chapitre V nous avons décrit la plate-forme développée au cours de cette thèse pour assembler les différents outils décrits. Nous pouvons ainsi partir des images issues du microscope confocale et aller jusqu'à la simulation de flux sur le méristème observé. Nous avons à cette occasion étudié différentes techniques liées au mélange d'interface graphique et d'interface langagière, les problèmes architecturaux que posent la création d'une plate-forme multi-langage, avec la nécessité de modulariser au mieux l'application pour qu'elle reste maintenable. Nous avons abordé le problème de l'extensibilité et de l'extractibilité des composants dans un environnement bi-langage (C++/Python), et les techniques de développement dites de "programmation générique" qui étaient utilisables dans ce cadre. Le logiciel a été intégralement placé sous licence GPL (GPL) pour contribuer à la communauté qui nous a fournit tant d'outils.

Cette thèse aura été l'occasion de combiner l'usage de la biologie du développement et des mathématiques, et plus particulièrement de l'informatique. D'un côté, l'informatique a permis d'extraire de l'information de données biologiques, de tester des hypothèses pour vérifier leur vraisemblance, de proposer de nouvelles études à la biologie. De l'autre, l'étude du méristème a soulevé des questions nécessitant de nouveaux développements mathématiques et informatiques. Ce travail a ainsi été l'occasion de travailler sur la communication inter-disciplinaire, avec la nécessité de définir un vocabulaire commun qui, au final, permet de s'exprimer sur les problèmes informatiques et les problèmes biologiques au sein d'une équipe mixte.

Perspectives

Pendant cette thèse, nous avons développé quatre axes de recherche centrés sur l'étude du méristème apical caulinaire. Dans chacun de ces axes, de nouvelles questions sont apparues aux cours de nos recherches.

Analyse et reconstruction géométrique. Plusieurs équipes ont travaillé sur ce problème spécifique, chacune partant de données de natures différentes (notamment, Dumais et Kwiatkowska (2001) sur le méristème caulinaire observé par microscopie électronique à balayage et Haseloff (2003) sur le méristème racinaire observé par microscopie confocale). Une question récurrente concerne la reconstruction 4D complète, donc une reconstruction volumique avec suivi dans le temps. La principale difficulté viendra probablement des contraintes de la mesure pour perturber la plante au minimum. Ainsi, il n'est pas possible de se placer dans ces conditions optimales de mesure sans tuer la plante quasi-systématiquement. Or, pour la réalisation d'un suivi, la plante doit subir plusieurs expositions au laser du microscope. Par contre, l'obtention d'un tel outil permettrait de réaliser des mesures précises de la croissance cellulaire dans les différentes assises cellulaire.

Une fois obtenues les reconstructions 4D, il sera ensuite nécessaire de développer les outils d'analyse adaptés. D'un côté les études statistiques, par exemple sur les parois cellulaires, devront tirer partie des recherches sur les statistiques spatiales et cycliques, mais en les adaptant pour fonctionner sur les effectif relativement réduit. D'un autre côté, des analyses structurelles doivent être mises en place, mais en prenant en compte les erreurs de mesures car elles sont loin d'être négligeables.

Étude des flux d'auxine dans les méristèmes réels. L'étude que nous avons menée semble indiquer que l'auxine se concentre dans le centre du méristème mais que celui-ci n'y est pas réactif. Nous avons déjà obtenu un certain nombre de confirmations biologiques de ce phénomène, mais nous envisageons plusieurs autres expériences sur la base de ce résultat. Une première possibilité serait d'utiliser de l'auxine radioactive. En introduisant de l'auxine radioactive à la base du méristème dans la plante adulte, il devrait être possible de voir comment celle-ci se répartit et notamment s'il y a une accumulation dans le centre. Une autre expérience nécessiterait de pouvoir dégrader l'auxine conditionnellement à l'activation d'un gène et idéalement de manière inductible. Ainsi, il serait possible de dégrader l'auxine dans la zone centrale du méristème dans la plante adulte. Si l'accumulation d'auxine est nécessaire, le phénotype devrait être semblable à celui d'un mutant *pin1*.

Un autre axe de recherche s'orienterait plutôt vers les causes du positionnement de la protéine PIN1. Nous avons vu au chapitre I que la protéine PID pourrait orienter PIN1 en agissant comme un interrupteur : au dessus d'une certaine concentration, PIN1 est orientée vers l'apex caulinaire, alors qu'en dessous elle est orientée vers la base. Cette observation soulève plusieurs questions. Est-ce que la concentration en protéine PID suffit à expliquer l'orientation de PIN1 ? Si oui, comment la concentration en protéine PID est-elle régulée ? Si non, quel autre facteur peut intervenir dans l'orientation de la protéine PIN1 ?

Simulation dynamique du méristème. Le système que nous avons développé jusqu'à présent fonctionne partiellement. Vraisemblablement, le modèle mécanique que nous avons mis en place est trop simple pour permettre la stabilisation d'un motif phyllotaxique. Une piste pour améliorer le modèle existant consiste donc à remplacer le modèle que nous avons créé par un modèle beaucoup plus proche d'un tissus cellulaire végétal. Notamment, la conservation du voisinage et la déformation de la surface dans un espace tri-dimensionnel sont des pistes intéressantes.

Ensuite, un problème avec notre modèle actuel concerne les conditions aux bords du méristème virtuel. En effet, avec le modèle actuel, les primordia disparaissent instantanément de la simulation s'ils sortent des bornes du méristème virtuel, ce qui induit une discontinuité forte dans les concentrations d'auxine. Il serait alors intéressant d'étudier un moyen de garder une mémoire des primordia sortis et de calculer leur impact sur la partie simulée du méristème.

Enfin, pour aller au-delà de ce que nous proposons dans cette thèse, une possibilité serait de représenter non plus la couche épidermique du méristème mais le dôme dans son intégralité. Une telle représentation nécessite un modèle tri-dimensionnel des cellules, ce qui implique des travaux d'une part pour modifier de façon cohérente une telle structure, mais aussi pour représenter le résultat d'une simulation sur un support 2D (écran, papier...).

Développements logiciels futures. Le projet se déroulant au sein de l'équipe Virtual Plants à l'origine du projet ALEA (Atelier Logiciel pour l'Écophysioloqie et l'Architecture des plantes), Merrysim et Merryproj intégreront la plate-forme ALEA. Étant donnés les choix réalisés sur l'architecture de la plate-forme ALEA (i.e. avec une architecture en

Python), il sera nécessaire de redévelopper au moins en partie les interfaces de saisie existantes pour les intégrer complètement à Python et à l'interface graphique de ALEA. En ce qui concerne la partie fonctionnelle, étant indépendante de toute interface graphique et déjà utilisable directement en Python, elle devrait être intégrable sans modification ou presque dans la plate-forme ALEA.

Le développement des nouveaux outils pourront tirer partie de la plate-forme ALEA, à la fois pour utiliser les outils proposés par la plate-forme et aussi comme médium de diffusion dans la communauté scientifique. Ainsi, la plate-forme ALEA intégrera un ensemble de modules adaptés à l'étude et la simulation de la morphogénèse à l'échelle tissulaire.

Bibliographie

Abrahams D. : Building hybrid systems with boost.python. Rapport technique, Boost Consulting, 2003. URL `http://www.boost-consulting.com/writing/bpl.html`.

Adler I., Barabe D. et Jean R. V. : A history of the study of phyllotaxis. *Annals of Botany*, 80(3):231–244, septembre 1997.

Aida M., Ishida T., Fukaki H., Fujisawa H. et Tasaka M. : Genes involved in organ separation in *Arabidopsis*: an analysis of the *cup-shaped cotyledon* mutant. *Plant Cell*, 9:841–857, 1997.

Aida M., Ishida T. et Tasaka M. : Shoot apical meristem and cotyledon formation during arabidopsis embryogenesis: interaction among the *CUP-SHAPED COTYLEDON* and *SHOOT MERISTEMLESS* genes. *Development*, 126:1563–1570, 1999.

Airy H. : On leaf-arrangement. *Proceedings of the Royal Society of London*, 21:176–179, février 1873.

Airy H. : On leaf-arrangement. *Proceedings of the Royal Society of London*, 22:298–307, avril 1874.

Aurenhammer F. : Voronoi diagrams: a survey of a fundamental geometric data structure. *Computing Surveys*, 23(3):345–406, septembre 1991. ISSN 0360-0300.

Avsian-Kretchmer O., Cheng J.-C., Chen L., Moctezuma E. et Z. R. S. : Indole acetic acid distribution coincides with vascular differenciation pattern during arabidospsis leaf ontogeny. *Plant Physiology*, 130:199–209, septembre 2002.

Barber C. B., Dobkin D. P. et Huhdanpaa H. : The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software*, 22(4):469–483, décembre 1996. ISSN 0098-3500. URL `http://www.qhull.org/`.

Bartel B. : AUXIN BIOSYNTHESIS. *Annual review of Plant Physiology and Plant Molecular biology*, 48:51–66, juin 1997.

Barton M. et Poethig R. : Formation of the shoot apical meristem in itshape Arabidopsis thaliana: an analysis of development in the wild-type and in the *shoot meristemless* mutant. *Development*, 119(3):823–831, novembre 1993.

Benkova E., Michniewicz M., Sauer M., Teichmann T., Seifertova D., Jurgens G. et Friml J. : Local, efflux-dependent auxin gradients as a common module for plant organ formation. *Cell*, 115:591–602, 2003.

Bennett M. J., Marchant A., Green H. G., May S. T., Ward S. P., Millner P. A., Walker A. R., Schulz B. et Feldmann K. A. : Arabidopsis *AUX1* gene: a permease-like regulator of root gravitropism. *Science*, 273(5277):948–950, août 1996.

Bennett S. R., Alvarez J., Bossinger G. et Smyth D. R. : Morphogenesis in *pinoid* mutants of *Arabidopsis thaliana*. *The Plant Journal*, 8(4):505–520, octobre 1995.

Boas M. L. : *Methematical methods in the physical sciences*, chapitre Ordinary differential equations, pages 337–381. Wiley, 2 édition, 1983.

Bodenstein L. : A dynamic simulation model of tissue growth and cell patterning. *Cell Differentiation*, 19(1):19–33, juillet 1986.

Boissonnat J.-D. et Yvinec M. : *Géométrie algorithmique*. Ediscience international, 1995.

Brand U., Fletcher J. C., Hobe M., Meyerowitz E. M. et Simon R. : Dependence of stem cell fate in *Arabidopsis* on a feedback loop regulated by *CLV3* activity. *Science*, 289:617–619, 2000.

Braun A. : Vergleichende Untersuchung über die Ordnung der Schuppen an den Tannenzapfen : als Einleitung zur Untersuchung der Blattstellung überhaupt. *Nova Acta Academiae Caesareae Leopoldino Carolinae Germanicae Naturae Curiosorum*, 15:195–402, 1831.

Braun A. : Dr. Schimper's Vortrage über die Moglichkeit eines wissenschaftlichen Verstandnisses der Blattstellung ... Flora. *Iena*, 18:145–191, 1835.

Bravais L. et Bravais A. : Essai sur la disposition des feuilles curvisériées. *Annales des Sciences Naturelles*, 7:42–110, 1837a.

Bravais L. et Bravais A. : Essai sur la disposition symétrique des inflorescences. *Annales des Sciences Naturelles*, 8:11–42, 1837b.

Bravais L. et Bravais A. : Essaie sur la disposition des feuilles rectisériées. *Annales des Sciences Naturelles*, 12(2):5–14 & 65–77, 1839.

Byrne M. E., Barley R., Curtis M., Arroyo J. M., Dunham M., Hudson A. et Martienssen R. A. : *Asymmetric leaves1* mediates leaf patterning and stem cell function in arabidopsis. *Nature*, 408:967–971, 2000.

Chapman J. M. et Perry R. : A diffusion model of phyllotaxis. *Annals of Botany*, 60:377–389, 1987.

Chuck G., Lincoln C. et Hake S. : *KNAT1* induces lobed leaves with ectopic meristems when overexpressed in arabidopsis. *Plant Cell*, 8:1277–1289, 1996.

Clark S. E., Jacobsen S. E., Levin J. Z. et Meyerowitz E. M. : The *CLAVATA* and *SHOOT MERISTEMLESS* loci competitively regulate meristem activity in *Arabidopsis*. *Development*, 122(5):1567–1575, mai 1996.

Clark S. E., Running M. P. et Meyerowitz E. M. : CLAVATA1, a regulator of meristem and flower development in *Arabidopsis*. *Development*, 119(2):397–418, octobre 1993.

Clark S. E., Running M. P. et Meyerowitz E. M. : CLAVATA3 is a specific regulator of shoot and floral meristem development affecting the same processes as CLAVATA1. *Development*, 121(7):2057–2067, juillet 1995.

Clark S. : Cell signalling at the shoot meristem. *Plant, Cell and Environment*, pages 276–84, 2001.

Couder Y. : Initial transitions, order and disorder in phyllotactic patterns: the ontogeny of *Hellianthus annuus*. A case study. 67(2):129–150, 1998.

Cummings F. W. et Strickland J. C. : A model of phyllotaxis. *Journal of Theoretical Biology*, 192:531–544, 1998.

Davies P. J. : *Plant Hormones: physiology, biochemistry and molecular biology*. Kluwer Academic Press, 1995.

Delbarre A., Muller P., Imhoff V. et Guern J. : Comparison of mechanisms controlling uptake and accumulation of 2,4-dichlorophenoxy acetic acid, naphthalene-1-acetic acid, and indole-3-acetic acid in suspension-cultured tobacco cells. *Planta*, 198 (4):532–541, avril 1996.

Deshaies R. J. : SCF and Cullin/Ring H2-based ubiquitin ligases. *Annu Rev Cell Dev Biol*, 15:435–467, novembre 1999.

Dharmasiri N., Dharmasiri S. et Estelle M. : The F-box protein TIR1 is an auxin receptor. *Nature*, 435(7041):441–445, mai 2005.

Douady S. et Couder Y. : Phyllotaxis as a physical self-organized growth process. *Physical Review Letters*, 68(13):2098–2101, mars 1992.

Douady S. et Couder Y. : Phyllotaxis as a dynamical self organizing process. part i: The spiral modes resulting from time-periodic iterations. *Journal of Theoretical Biology*, 178:255–274, 1996a.

Douady S. et Couder Y. : Phyllotaxis as a dynamical self organizing process. part ii: Tthe spontaneous formation of a periodicity and the coexistence of spiral and whorled patterns. *Journal of Theoretical Biology*, 178:275–294, 1996b.

Douady S. et Couder Y. : Phyllotaxis as a dynamical self organizing process. part iii: The simulation of the transient regimes of ontogeny. *Journal of Theoretical Biology*, 178:275–294, 1996c.

Dumais J. et Kwiatkowska D. : Analysis of surface growth in shoot apices. *The Plant Journal*, 31(2):229–241, 2001.

Edlund A., Eklöf S., Sundberg B., Moritz T. et Sandberg G. : A microscale technique for gas chromatography-mass spectrometry measurement of picogram amounts of indole-3-acetic acid in plant tissues. *Plant Physiology*, 108:1043–1047, 1995.

Elliott R. C., Betzner A., Huttner E., Oakes M. P., Tucker W. Q., Gerentes D., Perez P. et Smyth D. R. : *AINTEGUMENTA*, an *APETALA2*-like gene of *Arabidopsis* with pleiotropic roles in ovule development and floral organ growth. *Plant Cell*, 8:155–168, 1996.

Endrizzi K., Moussian B., Haecker A., Levin J. Z. et Laux T. : The *SHOOT MERISTEMLESS* gene is required for maintenance of undifferentiated cells in arabidopsis shoot meristems and acts at a different regulatory level than the meristem genes *WUSCHEL* and *ZWILLE*. *The Plant Journal*, 10:967–979, 1996.

Epstein E. et Ludwig-Müller J. : Indole-3-butyric acid in plants: occurrence, synthesis, metabolism and transport. *Plant Physiology*, 88:382–389, 1993.

Federl P. et Prusinkiewicz P. : Solving differential equations in developmental models of multicellular structures expressed using L-systems. *Dans* Bubak M., Albada G. v., Sloot P. et Dongarra J., éditeurs : *Proceedings of Computational Science. ICCS 2004*, pages 65–72. Springer, juin 2004.

Fletcher J. C., Brand U., Running M. P., Simon R. et Meyerowitz E. M. : Signaling of cell fate decisions by *CLAVATA3* in arabidopsis shoot meristems. *Science*, 283:1911–1914, 1999.

Friml J., Yang X., Michniewicz M., Weijers D., Quint A., Tietz O., Benjamins R., Ouwerkerk P. B. F., Ljung K., Sandberg G., Hooykaas P. J. J., Palme K. et Offringa R. : A PINOID-dependent binary switch in apical-basal PIN polar targeting directs auxin efflux. *Science*, 306(5697):862–865, octobre 2004.

Galweiler L., Guan C., Muller A., Wisman E., Mendgen K., Yephremov A. et Palme K. : Regulation of polar auxin transport by atpin1 in arabidopsis vascular tissue. *Science*, 282:2226–30, 1998.

Giavitto J.-L., Godin C., Michel O. et Prusinkiewicz P. : Computational models for integrative and developmental biology. *Dans Colloque Modélisation et simulation de processus biologiques dans le contexte de la génomique*, page 43, mars 2002.

Giavitto J.-L. et Michel O. : MGS: a programming language for the transformations of topological collections. Rapport technique 61–2001, LaMI, mai 2001a.

Giavitto J.-L. et Michel O. : MGS: a ruled-based language for complex objects and collections. *Electronic Notes in Theoretical Computer Science*, 59(4), 2001b.

Giavitto J.-L. et Michel O. : Data structure as topological spaces. *Dans 3rd International Conference on Unconventional Models of Computation (UMC'02)*, pages 137–150. Springer-Verlag, octobre 2002.

GPL. The GPL licence can be downloaded at the URL: `http://www.gnu.org/licenses/gpl.html`.

Grandjean O., Vernoux T., Laufs P., Belcram K., Mizukami Y. et Traas J. : In vivo analysis of cell division, cell growth, and differentiation at the shoot apical meristem in arabidopsis. *Plant Cell*, 16:74–87, janvier 2004.

Gray W. M., Kepinski S., Rouse D., Leyser O. et Estelle M. : Auxin regulates SCF-TIR1-dependent degradation of AUX/IAA proteins. *Nature*, 414(6861):271–276, novembre 2001.

Gray W. M. et Estelle M. : Function of the ubiquitin-proteasome pathway in auxin response. *Trends in Biochemical Sciences*, 25(3):133–138, mars 2000.

Green P. B. : Pattern formation in shoots: a likely role for minimal energy configurations of the tunica. *International Journal of Plant Science*, 153(3):S59–S75, 1992.

Green P. B. : Expression of pattern in plants: combining molecular and calculus-based biophysical paradigms. *American Journal of Botany*, 86(8):1059–1076, août 1999.

Green P. B., Steele C. S. et Rennich S. C. : Phyllotactic patterns: a biophysical mechanism for their origin. *Annals of Botany*, 77(5):515–528, mai 1996.

Greenfield P., Miller J. T., Hsu J.-c. et White R. L. : numarray: A new scientific array package for python. *Dans PyCon*, mars 2003.

Hamant O., Nogué F., Belles-Boix E., Jublot D., Grandjean O., Traas J. et Pautot V. : The KNAT2 homeodomain protein interacts with ethylene and cytokinin signaling. *Plant Physiology*, 130(2):657–65, octobre 2002.

Haseloff J. : Old botanical techniques for new microscopes. *BioTechniques*, 34(6):1174–1182, juin 2003.

Heijmans H., Nacken P., Toet A. et Luc V. : Graph morphology. *Journal of visual communication and image representation*, 3(1):24–38, mars 1992.

Hobbie L. et Estelle M. : Genetic aproaches to auxin action. *Plant, Cell and Environment*, 17:525–540, 1994.

Hoff III K. E., Keyser J., Lin M., Manocha D. et Culver T. : Fast computation of generalized Voronoi diagrams using graphics hardware. *Computer Graphics*, 33(Annual Conference Series):277–286, 1999.

Hofmeister W. : *Handbuch der Physiologishen Botanik*, volume 1, chapitre Allgemeine Morphologie der Gewachse, pages 405–664. Engelmann, Leipzig, 1868.

Honda H. : Description of cellular patterns by dirichlet domains: The two-dimensional case. *Journal of Theoretical Biology*, 72:523–543, 1978.

Honda H. : *International review of cytology*, volume 81, chapitre Geometrical Models for Cells in Tissues, pages 191–247. Academic Press Inc, 1983.

Honda H. : Geometrical models for cells in tissues. *International Review of Cytology*, 81:191–248, 1994.

Ishida T., Aida M., Takada S. et Tasaka M. : Involvement of *CUP-SHAPED COTYLEDON* genes in gynoecium and ovule development in *arabidopsis* thaliana. *Plant Cell Physiology*, 41:60–67, 2000.

Jean R. V. : *Croissance végétale et morphogénèse*. Masson, Presses de l'Université du Québec, 1983.

Jeong S., Trotochaud A. E. et Clark S. E. : The arabidopsis *CLAVATA2* gene encodes a receptor-like protein required for the stability of the CLAVATA1 receptor-like kinase. *Plant Cell*, 11:1925–1933, 1999.

Jones A. M. : Auxin transport: down and out and up again. *Science*, 282(5397):2201–2202, décembre 1998.

Jones E., Oliphant T., Peterson P. et others : SciPy: Open source scientific tools for Python, 2001. URL `http://www.scipy.org/`.

Jönsson H., Heisler M., Reddy G. V., Agrawal V., Gor V., Shapiro B. E., Mjolsness E. et Meyerowitz E. M. : Modeling the organization of the *WUSCHEL* expression domain in the shoot apical meristem. *Bioinformatics*, 21 Suppl 1:i232–i240, juin 2005.

Kaplan D. R. et Hagemann W. : The relationship of cell and organism in vascular plants: are cells the building blocks of plant form? *Bioscience*, 41:693–703, 1991.

Kappen C. : Analysis of a complete homeobox gene repertoire: implications for the evolution of diversity. *Proceedings of the National Academy of Science of the USA*, 97:4481–4486, 2000.

Kayes J. M. et Clark S. E. : CLAVATA2, a regulator of meristem and organ development in *Arabidopsis*. *Development*, 125(19):3843–3851, octobre 1998.

Kepinski S. et Leyser O. : The *Arabidopsis* F-box protein TIR1 is an auxin receptor. *Nature*, 435:446–451, mai 2005.

Klucher K. M., Chow H., Reiser L. et Fischer R. L. : The *AINTEGUMENTA* gene of *Arabidopsis* required for ovule and female gametophyte development is related to the floral homeotic gene *APETALA2*. *Plant Cell*, 8:137–153, 1996.

Kniemeyer O. : Rule-based modelling with the xl/groimp software. *Dans* Schaub H., Detje F. et Brüggemann U., éditeurs : *The Logic of Artificial Life*, Proceedings of 6th GWAL, pages 56–65, avril 2004.

Krizek B. A. : Ectopic expression of *AINTEGUMENTA* in *Arabidopsis* plants results in increased growth of floral organs. *Developmental Genetics*, 25:224–236, 1999.

Kwiatkowska D. et Dumais J. : Growth and morphogenesis at the vegetative shoot apex of *Anagallis arvensis* L. *Journal of Experimental Botany*, 54(387):1585–1595, juin 2003.

Lantin M. : Computer simulations of developmental processes. Rapport technique, Simon Fraser University, 1996. URL `http://www.marialantin.com/research/papers/depth.pdf`.

Laufs P., Grandjean O., Jonak C., Kieû K. et Traas J. : Cellular parameters of the shoot apical meristem in arabidopsis. *Plant Cell*, 10:1375–1389, août 1998.

Laux T. : The stem cell concept in plants: A matter of debate. *Cell*, 113:281–283, 2003.

Laux T., Mayer K., Berger J. et Jürgens G. : The *WUSCHEL* gene is required for shoot and floral meristem integrity in *Arabidopsis*. *Development*, 122:87–96, 1996.

Lejeune-Dirichlet G. : Über die Reduction der positiven quadratischen Formen mit drei unbestimmten ganzen Zahlen. *Journal für die reine und angewandte Mathematik*, 40:209–234, 1850.

Lincoln C., Long J., Yamaguchi J., Serikawa K. et Hake S. : A *KNOTTED1*-like homeobox gene in arabidopsis is expressed in the vegetative meristem and dramatically alters leaf morphology when overexpressed in transgenic plants. *Plant Cell*, 6:1859–1876, 1994.

Lindenmayer A. : Mathematical models for cellular interactions in development. i. filaments with one-sided inputs. *Journal of Theoretical Biology*, 18(3):280–299, 1968.

Ljung K., Rishikesh P., Bhalerao et Sandberg G. : Sites and homeostatic control of auxin biosynthesis in arabidopsis during vegetative growth. *Journal of Theoretical Biology*, 28(4):465–474, 2001.

Lomax T., Muday G. et Rubery P. : *Plant hormones: physiology, biochemistry and molecular biology*, chapitre Auxin Transport, pages 509–530. Kluwer Academic Publishers, 1995.

Long J. A. et Barton M. K. : The development of apical embryonic pattern in *Arabidopsis*. *Development*, 125:3027–3035, 1998.

Long J. A., Moan E. I., Medford J. I. et Barton M. K. : A member of the KNOTTED class of homeodomain proteins encoded by the *STM* gene of arabidopsis. *Nature*, 379(66-69), janvier 1996.

Lück J. et Lück H. B. : Generation of 3-dimensional plant bodies by double wall map and stereomap systems. *Dans* Ehrig H., Nagl M. et Rozenberg G., éditeurs : *Graph-Grammars and Their Application to Computer Science*, volume 153 de *Lecture Notes in Computer Science*, pages 219–231. Springer, 1982. ISBN 3-540-12310-5.

Lyndon R. : Rates of cell division in the shoot apical meristem of pisum. *Annals of Botany*, 34:1–17, 1970.

Lyndon R. : *The shoot apical meristem.* Cambridge University Press, 1998.

Marchant A., Kargul J., May S. T., Muller P., Delbarre A., Perrot-Rechenmann C. et Bennett M. J. : AUX1 regulates root gravitropism in *Arabidopsis* by facilitating auxin uptake within root apical tissues. *EMBO J*, 18(8):2066–2073, Apr 1999.

Marquardt D. : An algorithm for least squares estimation of nonlinear parameters. *SIAM Journal of Applied Mathematic*, 11:431–441, 1963.

Matela R. J. et Fletterick R. J. : A topological exchange model for cell self-sorting. *Journal of Theoretical Biology*, 76(4):403–414, février 1979.

Mayer F., K., Schoof H., Haecker A., Lenhard M., Jürgens G. et Laux T. : Role of *WUSCHEL* in regulating stem cell fate in the arabidopsis shoot meristem. *Cell*, 95:805–815, 1998.

Meinhardt H. : *The algorithmic beauty of sea shells.* The Virtual Laboratory. Springer, 3ème édition, 2003a.

Meinhardt H. : Complex pattern formation by a self-destabilization of established patterns: chemotactic orientation and phyllotaxis as examples. *Comptes Rendus Biologies*, 326(2):223–237, février 2003b.

Meyerowitz E. M. : Plant development: local control, global patterning. *Current Opinion in Genetic and Development*, 6(4):475–479, août 1996.

Mitchison G. J. : The polar transport of auxin and vein pattern in plants. *Philosophical Transactions of the Royal Society of London. B. Biological Sciences*, 295:461–271, 1981.

Mizukami Y. et Fischer R. L. : Plant organ size control: *AINTEGUMENTA* regulates growth and cell numbers during organogenesis. *Proceedings of the National Academy of Science of the USA*, 97:942–947, 2000.

Moctezuma E. et Lewis J. F. : Auxin redistribution upwards in graviresponding gynophores of the peanut plant. *Planta*, 209:180–186, 1999.

Moré J. J., Garbow B. S. et Hillstrom K. E. : User guide for MINPACK-1. Rapport technique ANL-80-74, Argonne National Laboratory, Argonne, IL, USA, août 1980.

Morris D., Rubery P., Jarman J. et Sabater M. : Effects of inhibitors of protein synthesis on transmembrane auxin transport in *Cucurbita pepo* L. hypocotyl segments. *Journal of Experimental Botany*, 42:773–?783, 1991.

Nougarède A. : Experimental cytology of the shoot apical cells during vegetative growth and flowering. *International Review of Cytology*, 21:203–351, 1967.

Ori N., Eshed Y., Chuck G., Bowman J. L. et Hake S. : Mechanisms that control *KNOX* gene expression in the arabidopsis shoot. *Development*, 127:5523–5532, 2000.

Pautot V., Dockx J., Hamant O., Kronenberger J., Grandjean O., Jublot D. et Traas J. : *KNAT2*, Evidence for a link between Knotted-like genes and carpel development. *The Plant Cell*, 13(8):1719–1734, août 2001.

Plantefol L. : *La théorie des hélices foliaires multiples*. Fondements d'une théorie phyllotaxique nouvelle. Masson & Cie, 1948.

Porter T. et Duff T. : Composing digital images. *SIGGRAPH Comput. Graph.*, 18 (3):253–259, 1984. ISSN 0097-8930.

Pradal C., Dones N., Godin C., Barbier de Reuille P., Boudon F., Adam B. et Sinoquet H. : ALEA: A software for integrating analysis and simulation tools for 3D architecture and ecophysiology. *Dans* Godin C., Hanan J., Kurth W., Lacointe A., Takenaka A., Prusinkiewicz P., DeJong T., Beveridge C. et Andrieu B., éditeurs : *4th International Workshop on Functional-Structural Plant Models*, page 406, Montpellier, France, juin 2004.

Press W. H., Flannery B. P., Teukolsky S. A. et Vetterling W. T. : *Numerical Recipes: The Art of Scientific Computing*. Cambridge University Press, Cambridge (UK) and New York, 2nd édition, 1992. ISBN 0-521-43064-X.

Prusinkiewicz P. et Lindenmayer A. : *The algorithmic beauty of plants*. The Virtual Laboratory. Springer, 1990. URL `http://algorithmicbotany.org/papers/#abop`.

Raven J. : Transport of indolacetic acid in plant cells in relation to pH and electrical gradients, and its significance for polar IAA transport. *New Phytologist*, 74:163–172, 1975.

Reddy G. V., Heisler M. G., Ehrhardt D. W. et Meyerowitz E. M. : Real-time lineage analysis reveals oriented cell divisions associated with morphogenesis at the shoot apex of *Arabidopsis thaliana*. *Development*, 131:4225–37, septembre 2004.

Reinhardt D., Frenz M., Mandel T. et Kuhlemeier C. : Microsurgical and laser ablation analysis of interactions between the zones and layers of the tomato shoot apical meristem. *Development*, 130:4073–4083, 2003a.

Reinhardt D., Mandel T. et Kuhlemeier C. : Auxin regulates the initiation and radial position of plant lateral organs. *Plant Cell*, 12(4):507–518, avril 2000.

Reinhardt D., Pesce E.-R., Stieger P., Mandel T., Baltensperger K., Bennett M., Traas J., Friml J. et Kuhlemeier C. : Regulation of phyllotaxis by polar auxin transport. *Nature*, 426:255–260, novembre 2003b.

Rempt B. : *GUI programming with Python: Qt edition*. OpenDocs LLC, 2001. URL `http://www.opendocspublishing.com/pyqt/index.lxp`.

Rogg L. E. et Bartel B. : Auxin signaling: derepression through regulated proteolysis. *Developmental Cell*, 1(5):595–604, novembre 2001.

Rubery P. et Sheldrake A. : Carrier-mediated auxin transport. *Planta*, 118(2):101–121, juin 1974.

Schimper K. F. : Beschreibung des Symphytum Zeyheri und seiner zwei deutschen Verwandten der *S . Bulborum* Schimp und *S . Tuberosum. Jacq . Geiger 's Mag . für Pharm.*, 29:1–92, 1830.

Schneeberger R., Tsiantis M., Freeling M. et Langdale J. A. : The *rough sheath2* gene negatively regulates homeobox gene expression during maize leaf development. *Development*, 125:2857–2865, 1998.

Schoof H., Lenhard M., Haecker A., F. M. K., G. J. et Laux T. : The stem cell population of arabidopsis shoot meristems in maintained by a regulatory loop between the *CLAVATA* and *WUSCHEL* genes. *Cell*, 100(6):635–644, 2000.

Semiarti E., Ueno Y., Tsukaya H., Iwakawa H., Machida C. et Machida Y. : The *ASYMMETRIC LEAVES 2* gene of arabidopsis thaliana regulates formation of a symmetric lamina, establishment of venation and repression of meristem-related homeobox genes in leaves. *Development*, 128:1771–1783, 2001.

Serra J. et Vincent L. : An overview of morphological filtering. *Circuits, Systems and Signal Processing*, 11(1):47–108, janvier 1992.

Shipman P. D. et Newell A. C. : Phyllotactic patterns on plants. *Physical Review Letters*, 92(16):168102, avril 2004.

Shipman P. D. et Newell A. C. : Polygonal planforms and phyllotaxis on plants. *Journal of Theoretical Biology*, 236(2):154–197, septembre 2005.

Siek J. et Lumsdaine A. : Concept checking: Binding parametric polymorphism in C++. *Dans First Workshop on C++ Template Programming, Erfurt, Germany*, octobre 2000.

Siek J. G., Lee L.-Q. et Lumsdaine A. : *Boost Graph Library, The User Guide and Reference Manual*. Addison Wesley Professional, 1st édition, 2001.

Smart J., Hock K. et Stefan C. : *Cross-Platform GUI Programming with wxWidgets*. Prentice Hall, 2005.

Snow M. et Snow R. : Auxin and leaf formation. *New Phytologist*, 36:1–18, 1937.

Souer E., van Houwelingen A., Kloos D., Mol J. et Koes R. : The *NO APICAL MERISTEM* gene of Petunia is required for pattern formation in embryos and flowers and is expressed at meristem and primordia boundaries. *Cell*, 85:159–170, 1996.

Steeves T. et Sussex I. : *Patterns in Plant Development*. Cambridge University Press, 2 édition, 1989.

Stieger P. A., Reinhardt D. et Kuhlemeier C. : The auxin influx carrier is essential for correct leaf positioning. *The Plant Journal*, 32(4):509–517, novembre 2002.

Takada S., Hibara K.-I., Ishida T. et Tasaka M. : The cup-shaped cotyledon1 gene of arabidopsis regulates shoot apical meristem formation. *Development*, 128:1127–1135, 2001.

Thomas C., Bronner R., Molinier J., Prinsen E., van Onckelen H. et Hahne G. : Immuno-cytochemical localization of indole-3-acetic acid during induction of somatic embryogenesis in cultured sunflower embryos. *Planta*, 215:577–583, 2002.

Timmermans M. C. P., Hudson A., Becraft P. W. et Nelson T. : ROUGH SHEATH 2: a Myb protein that represses knox homeobox genes in maize lateral organ primordia. *Science*, 284:151–153, 1999.

Tiwari S. B., Wang X. J., Hagen G. et Guilfoyle T. J. : AUX/IAA proteins are active repressors, and their stability and activity are modulated by auxin. *Plant Cell*, 13(12):2809–2822, décembre 2001.

Tooke F. et Battey N. : Models of shoot apical meristem function. *New Phytologist*, 159(1):37–52, juillet 2003.

Traas J. et Doonan J. : Cellular basis of shoot apical meristem function. *International Review of Cytology*, 208:161–206., 2001.

Traas J. : Unpublished observations.

Troll W. : *Vergleichende Morphologie der höheren Pflanzen*. Borntraeger, 1937.

Trolltech : Qt 3.3 whitepaper. URL `http://www.trolltech.com/pdf/whitepapers/qt33-whitepaper-a4.pdf`.

Trotochaud A. E., Hao T., Wu G., Yang Z. et Clark S. E. : The CLAVATA1 receptor-like kinase requires CLAVATA3 for its assembly into a signaling complex that includes KAPP and a Rho-related protein. *Plant Cell*, 11:393–406, 1999.

Tsiantis M. : Control of shoot cell fate: beyond homeoboxes. *Plant Cell*, 13:733–738, 2001.

Tsiantis M., Schneeberger R., Golz J. F., Freeling M. et Langdale J. A. : The maize *ROUGH SHEATH2* gene and leaf development programs in monocot and dicot plants. *Science*, 284:154–156, 1999.

Tuominen H., Ostin A., Sandberg G. et Sundberg B. : A novel metabolic pathway for indole-3-acetic acid in apical shoots of *Populus tremula* (L.) × *Populus tremuloides* (Michx.). *Plant Physiol*, 106(4):1511–1520, décembre 1994.

Ulmasov T., Hagen G. et Guilfoyle T. J. : ARF1, a transcription factor that binds to auxin response elements. *Science*, 276(5320):1865–1868, juin 1997.

Van den Berg C., Willemsen V., Hage W., Weisbeek P. et Scheres B. : Cell fate in the arabidopsis root meristem determined by directional signalling. *Nature*, 378 (6552):62–5, 1995.

van Iterson G. : *Mathematische und Microscopisch Anatomische Studien über Blattstellungen, nebst Betraschungen über den Schalebau der Miliolinen.* Gustav-FischerVerlag, Iena, 1907.

van Rossum G. : Python language website, 2005. URL `http://www.python.org`.

Veen A. H. et Lindenmayer A. : Diffusion mechanism for phyllotaxis: theoretical physico-chemical and computer study. *Plant Physiology*, 60(1):127–139, juillet 1977.

Vernoux T. : *Différenciation cellulaire dans l'apex caulinaire d'Arabidopsis thaliana: rôle du transport polarisé de l'auxine.* Thèse de doctorat, Université de Paris-Sud U.F.R. scientifique d'Orsay, 2002.

Vernoux T., Kronenberger J., Grandjean O., Laufs P. et Traas J. : PIN-FORMED 1 regulates cell fate at the periphery of the shoot apical meristem. *Development*, 127:5157–5165, 2000.

Vincent L. : *Mathematical Morphology in Image Processing*, chapitre Morphological Algorithms, pages 255–288. Marcel-Decker, septembre 1992.

Vincent L. et Soille P. : Watersheds in digital spaces: An efficient algorithm based on immersion simulations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(6):583–598, juin 1991.

Vitha S., Baluska F., Mews M. et Volkmann D. : Immunofluorescence detection of f-actin on low melting point wax sections from plant tissues. *The Journal of Histochemistry and Cytochemistry*, 45(1):89–95, 1997. After embedding, the meristems were sectionned perpendicular to the main stems with a thickness of $12 - 15\mu$m. After labeling with anti-PIN1, the physical sections were viewed in the confocal microscope to obtain an optimal image of the labeling patterns. In some cases, a single section was sufficient to cover the entire dome of the meristem. In other cases, the patterns of two successive sections were combined to cover the dome.

Vogler H. et Kuhlemeier C. : Simple hormones but complex signalling. *Current Opinion in Plant Biology*, 6:51–6, 2003.

Voronoï G. F. : Nouvelles applications des paramètres continus à la théorie des formes quadratiques. deuxième mémoire: recherches sur les parallelloèdres primitifs. *Journal für die reine und angewandte Mathematik*, 134:198–287, 1908.

Vroemen C. W., Mordhorst A. P., Albrecht C., Kwaaitaal M. A. C. J. et de Vries S. C. : The CUP-SHAPED COTYLEDON3 gene is required for boundary and shoot meristem formation in Arabidopsis. *Plant Cell*, 15(7):1563–1577, Jul 2003.

Waites R., Selvadurai H. R., Oliver I. R. et Hudson A. : The *PHANTASTICA* gene encodes a Myb transcription factor involved in growth and dorsoventrality of lateral organs in *Antirrhinum*. *Cell*, 93:779–789, 1998.

Wardlaw C. : Evidence relating to the diffusion-reaction theory of morphogenesis. *New Phytologist*, 54:39–48, 1955a.

Wardlaw C. : Responses of fern apex to direct chemical treatments. *Nature*, 176:1098–1100, 1955b.

Webb R. H. : Confocal optical microscopy. *Reports of Progress in Physics*, 59:427–471, mars 1996.

website Boost : The Boost website. URL `http://boost.org/`.

website GNOME : The GNOME website. URL `http://www.gnome.org/`.

website GTK+ : The GTK+ website. URL `http://www.gtk.org/`.

website Gtkmm : The Gtkmm website. URL `http://www.gtkmm.org/`.

website KDE : The KDE website. URL `http://www.kde.org/`.

website OSI : The OSI website. URL `http://www.opensource.org/`.

website PIL : The Python Image Library website. URL `http://www.pythonware.com/products/pil/index.htm`.

website PyGTK : The PyGTK website. URL `http://www.pygtk.org/`.

website R : The R project website. URL `http://www.r-project.org/`.

website RPY : The RPy website. URL `http://rpy.sourceforge.net/`.

website SciPy : The SciPy website. URL `http://www.scipy.org/`.

website SIP : The SIP website. URL `http://www.riverbankcomputing.co.uk/sip/`.

website Swig : The Swig website. URL `http://www.swig.org/`.

website The Gimp : The Gimp website. URL `http://www.gimp.org/`.

website wxPython : The wxPython website. URL `http://www.wxpython.org/`.

website wxWidget : The wxWidgets website. URL `http://www.wxwidgets.org`.

Weigel D. et Jürgens G. : Stem cells that make stems. *Nature*, 6873(415):751–754, février 2002.

Wolff C. F. : *Theoria generationis*. Willhelm Engelmann, Leipzig, Germany, 1896.

Wu X., Weigel D. et Wigge P. : Signalling in plants by intercellular RNA and protein movement. *Genes & Development*, 16:151–158, 2002.

Xu C., Zipfel W., Shear J. B., Williams R. M. et Webb W. W. : Multiphoton fluorescence excitation: New spectral windows for biological nonlinear microscopy. *Proceedings of the National Academy of Science of the USA*, 93:10763–10768, octobre 1996.

Young D. A. : On the diffusion theory of phyllotaxis. *Journal of Theoretical Biology*, 71(3):421–432, avril 1978.

Glossaire de biologie

abaxial - Dirigé à l'opposé de la tige de la plante. Pour les feuilles, il s'agit de la face inférieure, celle qui fuit l'apex. Opposé de adaxial⋆.

adaxial - Dirigé vers la tige de la plante. Pour les feuilles, il s'agit de la face supérieure, celle qui est dirigée vers l'apex. Opposé de abaxial⋆.

AIA - Acide indole-3-acétique, forme la plus courante de l'auxine dans la plante sauvage.

allèle - Mutation particulière dans un gène. Les différents allèles d'un même gène sont souvent numérotés, on parlera par exemple de l'allèle *stm-1* décrit par Barton et Poethig (1993) et *stm-2* décrit par Clark *et al.* (1996) qui sont deux mutations de type perte de fonction sur le gène *STM*. On désigne par allèle fort les allèles dont le phénotype est le plus marqué et par allèle faible ceux dont le phénotype est le moins marqué.

angiosperme - Les angiospermes désignent l'ensemble des plantes à fleurs. Les angiospermes se divisent intégralement entre les dicotylédones⋆ et les monocotylédones⋆.

angle de divergence - Angle formé par deux organes successifs par rapport à l'axe de la tige.

anticlinal - Perpendiculaire à la surface du méristème.

bractée - Petite feuille de forme intermédiaire entre la feuille et le pétale, poussant à proximité de la fleur qu'elle recouvre en partie avant son éclosion.

caulinaire - Qui croît sur la tige. Du latin *caulis* : tige de la plante.

cellule-autonome - Qui ne dépend que de l'état interne de la cellule. Pour une action d'une cellule sur une autre on parle de processus non cellule-autonome.

chimère - Plante dont les cellules diffèrent pour un ou plusieurs marqueurs. Technique utilisée pour suivre la provenance des cellules.

cotylédon - Les cotylédons sont les feuilles primordiales constitutives de la graine, ils sont chargés de divers types de réserves : protéines, lipides, et sucres.

cytologie - Étude de la cellule, normale ou pathologique, notamment de sa morphologie, ses propriétés physiques, chimiques et physiologiques et son évolution.

dicotylédone - Plante dont l'embryon a deux cotylédons⋆. Voir angiosperme⋆.

ectopique - Qui ne se trouve pas à sa vrai place.

épistatique - Une mutation *ma* est dite épistatique sur une mutation *mb* si le phénotype du double mutant *ma mb* est identique au phénotype du mutant *ma*.

facteur de transcription - Protéine modifiant la transcription d'un gène.

gènes à homéoboîte - Les gènes à homéoboîte, d'abord découvers chez la drosophile, forment une famille de gènes caractérisés par un motif particulier : l'homéoboîte. Ils codent le plus souvent pour des facteurs de transcription (des protéines à homéodomaine) et sont des régulateurs essentiels du développement des eucaryotes multicellulaires (pour revue : Kappen, 2000).

histologie - Étude de la description microscopique des tissus animaux ou végétaux et des cellules qui les composent.

homologue - Deux gènes sont dits homologues lorsqu'ils appartiennent à la même espèce et ont des séquences similaires. Très souvent, des gènes homologues auront des fonctions proches.

hypocotyle - Partie de la plante située entre les cotylédons⋆ et la racine primaire.

immunomarquage - Technique d'observation des protéines dans un tissu. Le principe repose sur l'utilisation d'un anticorps ciblant une protéine connue (cet anticorps étant généralement fabriqué par génie génétique). Parfois, il est nécessaire de combiner une suite d'anticorps (chaque anticorps ciblant le complexe formé par l'anticorps précédent et sa cible). Le dernier anticorps de la suite étant couplé à la molécule fluorescente, il est possible d'observer sa position dans le tissu avec des techniques de microscopie à fluorescence. Si le ciblage est suffisamment précis, il est possible de considérer que la position de cet anticorps est identique à la position de la protéine dans le tissu. Il faut noter que cette technique nécessite une section physique du tissu, l'anticorps étant incapable de passer la membrane plasmique : le tissu est donc fixé pour l'observation.

kinase - Protéine ayant pour rôle de catalyser la phosphorylation de substrats, ce qui implique le plus souvent la catalyse de l'action d'autres protéines.

***KNOX* -** Famille de gènes à homéoboîte⋆ similaires aux gènes *KNOTTED* (i.e. *KNOTTED*-like homeobox).

ligase - Protéine ayant pour rôle de lier entre elles d'autre protéines.

médullaire - Qui forme la partie interne d'un organe ou qui s'y rapporte.

monocotylédone - Plante à graine qui, contrairement à une dicotylédone⋆, ne semble pas avoir de vrai cotylédon⋆. Voir angiosperme⋆.

orthologue - Deux gènes sont dits orthologues lorsqu'ils appartiennent à des espèces différentes mais ont des séquences similaires. Très souvent, des gènes orthologues auront des fonctions proches.

péricline - Parallèle à la surface du méristème.

phyllotaxie - Position des feuilles ou plus généralement des organes latéraux de la plante.

pluripotent - Se dit de cellules capable de donner naissance à plusieurs des types cellulaires d'un organisme.

stèle - Partie centrale des tiges et des racines.

symport - Transport de plusieurs substances dans la même direction.

tégument - Membrane protectrice qui entoure un organe végétal.

tropisme - Orientation, le plus souvent de la croissance, résultant de l'interaction avec l'environnement.

ubiquitination - C'est un processus de dégradation des protéines. Elles sont dans un premier temps "étiquetées" par un peptide de 70-80 acides aminés (l'ubiquitine*). Une fois qu'elles portent cette étiquette, elles vont être reconnues par de complexes d'enzymes qui la dégradent ensuite.

ubiquitine - Peptide de 70-80 acides aminés impliqué dans l'ubiquitination*.

Glossaire de mathématique et informatique

API - Application Programming Interface, désigne les symboles, les fonctions et les structures publiées par une bibliothèque ou un programme à destination de programmes tiers.

bruit additif - Un processus peut être considéré comme un bruit pour divers raisons, principalement si c'est un processus perturbant le processus étudié. On qualifiera le bruit d'additif si la perturbation correspond à l'addition d'un processus aléatoire.

chaîne - Ensemble ordonné d'élements, possédant donc un premier élément, un dernier élément, et un successeur unique pour chaque élément.

complétude - On parle de la complétude d'un ensemble d'hypothèse s'il n'en manque aucune pour répondre aux questions posées.

concept - Ensemble de contraintes qu'un type doit respecter pour être utilisé dans le contexte d'un algorithme générique. Ces contraintes consistent le plus souvent en un ensemble de méthodes ou de fonctions qui doivent être applicables sur les instances du type. Mais il y a souvent, associés à une méthode donnée, une contrainte sur la complexité en temps ou en espace (Siek et Lumsdaine, 2000).

délégation - Le design pattern* de la délégation permet d'appeler les méthodes définies dans un objet à partir d'un autre objet le contenant. L'objet contenant définit des méthodes qui appelleront simplement les méthodes de l'objet contenu. La délégation permet entre autre de choisir à l'exécution quel objet exécutera la méthode demandée.

dead-lock - Blocage complet du programme provoqué quand deux (ou plusieurs) thread (ou processus) attendent simultanément la fin des autres pour pouvoir continuer leur exécution.

design pattern - Ensemble de structures standards adaptées à une conception orientée objet et permettant de résoudre des problèmes régulièrement rencontrés dans les programmes informatiques.

dual - Qui est lié à un autre élément par une relation de correspondance réciproque.

entête - En C et C++ il est usuel de définir les fonctions et les structures utilisées dans des fichiers particuliers, appelés fichiers d'entêtes.

factory - Objet ou fonction créant des objets en fonction d'un ensemble de paramètres. Ce design pattern* propose de découpler complètement l'implémentation de l'objet et son initialisation. Il permet notamment de créer des objets différents en fonction des paramètres en tirant parti du polymorphisme*.

focus - Dans le vocabulaire des interface graphique, le focus désigne la gestion du composant actif à un moment donné. Ce sera notamment le composant qui recevra les événements claviers.

fraction continue - L'algorithme d'Euclide permet d'exprimer tout nombre réel positifs ω sous la forme : $\omega = a_0 + 1/(a_1 + 1/(a_2 + 1/(\dots)))$ où $a_0, a_1, a_2, \dots$ sont des entiers positifs. C'est ce qu'on appelle la fraction continue de ω.

JIT - "Just In Time (compiler)", compilateur à la volée. C'est une technique permettant de compiler très rapidement des expressions isolées d'un langage pour les évaluer dans le contexte courant d'exécution.

langage de script - Langage informatique pensé pour la réalisation de script*. Traditionnellement, les langages de scripts ne nécessitent pas de compilation explicite.

langage fonctionnel - Famille de langages informatiques basé sur le lambda-calcul. Ces langages se caractérisent par les possibilités de manipulation de fonctions comme objets et par le caractère immuable des objets utilisés.

licence libre - Licence autorisant l'utilisateur à copier, modifier et redistribuer librement le logiciel. Les seules contraintes possibles concernent la conservation de la licence dans les versions modifiées ou l'obligation de citer les différents auteurs dans les versions modifiées.

morphogène - Acteurs d'un processus de réaction-diffusion.

multi-ensemble - Ensemble dans lesquels plusieurs copies d'un même éléments peuvent coexistées.

patron - En C++, un patron de classe ou de fonction est une définition paramétrée par un ensemble de types ou de valeurs. Les patrons permettent de définir des objets ou des algorithmes génériques qui seront instanciés pour un jeu de paramètres à la compilation (et non à l'exécution).

pipe - Système de communication reposant sur la métaphore du dialogue dans un tuyaux. Un pipe est représenté par deux descripteurs de fichiers (les deux bouts du tuyaux), un servant à l'écriture et l'autre à la lecture. Les pipes peuvent être créés sur un système de fichier UNIX. L'accès est alors identique à celui d'un fichier standard et on parle de "pipe nommé".

polymorphisme - Dans les langages statiquement typés, le polymorphisme désigne la capacité d'une variable d'avoir un type dynamique différent de son type statique.

pré-processeur - Programme transformant un fichier source en un ou plusieurs autres fichiers sources, souvent pour automatiser la génération de code à partir de quelques directives.

proxy - Objet joue le rôle de relais pour un autre objet. Le but d'un proxy est variable, mais dans notre cas, il s'agit d'utiliser une référence vers un objet pour autoriser le polymorphisme* sans que l'utilisateur n'ait à gérer la mémoire, ni à travailler avec des pointeurs.

script - Ensembles relativement simple d'instructions permettant d'automatiser des tâches répétitives. Voir langage de script*.

widget - Contraction de "Window gadget", désigne un élément d'une interface graphique.

Annexe A

Concepts ALEA pour les classes de graphes

Similairement à la Boost Graph Library, le projet ALEA définit une hiérarchie de concepts pour les classes de graphes. Nous allons ici exposer cette hiérarchie et expliquer comment l'implémentation proposée par Merrysim permet à la fois de respecter cette hiérarchie et de travailler avec plusieurs classes d'arcs.

Le langage de communication du projet ALEA étant Python, c'est le langage de définition de ses concepts. L'équivalent des concepts C++ en Python correspondent ainsi à un ensemble de définitions de méthodes qui doivent être implémentées dans les classes accompagnées éventuellement de contraintes sur la complexité de l'algorithme implémenté. Dans le cas où la complexité est exprimée, n désigne le nombre de nœuds du graphe, m le nombre d'arcs et d l'ordre maximum parmi les nœuds du graphe.

Mais tout d'abord, pour décrire le prototype des méthodes, nous avons besoin d'une notation. Dans les définitions suivantes `[elmt]` signifie que `elmt` est optionnel, `elmt*` signifie que `elmt` est présent 0 ou plus de fois, `(elm1|elmt2)` indique un choix entre `elmt1` et `elmt2` et enfin les parenthèses servent à grouper des éléments. Dans la pseudo-grammaire que nous utilisons, l'axiome est un `type`.

`type` = `(classe|conteneur|fonction)`
: `classe` est le nom d'une classe (int,float,dict...), `fonction` désigne le prototype d'une fonction et `conteneur` est l'expression définissant une classe conteneur (i.e. dont le rôle est de structurer un ensemble de valeurs).

`fonction` = `fct([args])[ -> type]`
: définit une fonction dont le nom est `fct` dont les arguments sont `args` et qui renvoie une instance de `type` ou `None` si le type de retour n'est pas spécifié.

`args` = `arg(,arg)*`

`arg` = `[nom:]type`
: définit un argument dont le nom est `nom` et qui est une instance de `type`.

`conteneur` = `(classe of type2 | classe of type2:type3 | (type1[,type2]*))`
: où la première forme correspond à un conteneur simple (list, tuple, ensemble...), la seconde à un conteneur associatif (dictionnaire...) et la troisième à un conteneur (typiquement un tuple) dont la taille et le type des éléments sont connus.

Élément	Description
`is_valid() -> bool`	Vrai si le graphe est actuellement valide. $O(1)$
`vid`	Type de l'identifiant d'un nœud.
`eid`	Type de l'identifiant d'un arc.
`has_vertex(vtx:vid) -> bool`	Vrai si le nœud existe. On définira `__contains__` comme un alias vers cette méthode. $O(1)$
`has_edge(edge:eid) -> bool`	Vrai si l'arc existe. $O(1)$
`source(edge:eid) -> vid`	Renvoie le nœud source de l'arc. $O(1)$
`target(edge:eid) -> vid`	Renvoie le nœud but de l'arc. $O(1)$

TAB. A.1: **Concept de `Graph`.**

Les concepts définis sont :

`Graph :` concept principal que tout graphe doit respecter (voir table A.1) ;

`VertexGraph :` concept correspondant aux graphes permettant d'itérer sur ses nœuds (voir table A.2), ce concept dérive de celui de `Graph` ;

`EdgeGraph :` concept symétrique de `VertexGraph` mais pour les arcs (voir table A.3), ce concept dérive de celui de `Graph` ;

`MutableGraph :` concept correspondant aux graphes qui sont modifiables, on peut donc leur rajouter arcs ou nœuds (voir table A.4), ce concept dérive de ceux de `VertexGraph` et `EdgeGraph` ;

`CopyGraph :` concept correspondant aux graphes qui sont copiable (voir table A.5), ce concept dérive de celui de `Graph`.

Les concepts ALEA ne considèrent pas le cas des classes d'arcs multiples. Or il faut un moyen de sélectionner, pour toute opération sur les arcs, la classe d'arcs à utiliser.

Dans l'implémentation des graphes de Merrysim, les classes d'arcs sont identifiées par un entier. Ainsi, les fonctions sur les arcs admettent toutes un argument supplémentaire qui est la classe d'arcs. Mais ce choix ne respecte pas les concepts. Aussi, nous avons ajouté la notion de classe d'arcs courante. Ainsi, l'argument supplémentaire est facultatif et, s'il n'est pas précisé, c'est que la fonction est à appliquer à la classe d'arcs courante.

Il faut juste noter que seules deux méthodes n'ont pas l'argument supplémentaire pour désigner la classe d'arcs: les méthodes `edges()` et `nb_edges()`. Elle ne peuvent pas avoir d'argument supplémentaire, sinon leur signature entre en conflit avec, respectivement, les méthodes `edges(vid)` et `nb_edges(vid)`. Pour ces deux méthodes, le seul moyen de spécifier la classe d'arcs est donc de spécifier la classe d'arcs courante.

Élément	**Description**
`vertices() -> iter of vid`	Renvoie un itérateur sur l'ensemble des nœuds du graphe. On définira `__iter__` comme un alias vers cette méthode. $O(n)$
`nb_vertices() -> int`	Renvoie le nombre total de nœuds du graphe. On définira `__len__` comme un alias vers cette méthode. $O(1)$
`in_neighbors(vtx:vid) -> iter of vid`	Itérateur sur les voisins par un lien entrant d'un nœud. $O(d)$
`nb_in_neighbors(vtx:vid) -> int`	Nombre de voisins par un lien entrant d'un nœud. $O(1)$
`out_neighbors(vtx:vid) -> iter of vid`	Nombre de voisins par un lien sortant d'un nœud. $O(d)$
`nb_out_neighbors(vtx:vid) -> int`	Itérateur sur les voisins par un lien sortant d'un nœud. $O(1)$
`neighbors(vtx:vid) -> iter of vid`	Itérateur sur les voisins d'un nœud. $O(d)$
`nb_neighbors(vtx:vid) -> int`	Nombre de voisins d'un nœud. $O(1)$

TAB. A.2: **Concept `VertexGraph`.**

Élément	**Description**
`edge(source:vid,target:vid) -> iter of eid`	Renvoie un itérateur sur l'ensemble des arcs de source `source` et de but `but`. $O(1)$
`nb_edge(source:vid,target:vid) -> int`	Nombre d'arcs de source `source` et de but `target`. $O(1)$
`edges() -> iter of eid`	Renvoie un itérateur sur l'ensemble des arcs du graphe. $O(m)$
`nb_edges() -> int`	Renvoie le nombre total d'arcs. $O(1)$
`out_edges(vtx:vid) -> iter of eid`	Itérateur sur les arcs sortants d'un nœud. $O(d)$
`nb_out_edges(vtx:vid) -> int`	Nombre d'arcs sortants d'un nœud. $O(1)$
`in_edges(vtx:vid) -> iter of eid`	Itérateur sur les arcs entrants d'un nœud. $O(d)$
`nb_in_edges(vtx:vid) -> int`	Nombre d'arcs entrants d'un nœud. $O(1)$
`edges(vtx:vid) -> iter of eid`	Itérateur sur les arcs d'un nœud. $O(d)$
`nb_edges(vtx:vid) -> int`	Nombre d'arcs d'un nœud. $O(1)$

TAB. A.3: **Concept `EdgeGraph`.**

Élément	**Description**
`clear()`	Efface tous les nœuds et tous les arcs du graphe. $O(m+n)$
`clear_edges()`	Efface tous les arcs du graphe. $O(m)$
`add_vertex(vtx:vid)`	Ajoute un nœud d'identifiant `vtx` au graphe. $O(1)$
`add_vertex() -> vid`	Identique à la méthode précédente mais génère un identifiant de nœud valide. $O(1)$
`remove_vertex(vtx:vid)`	Enlève le nœud d'identifiant `vtx` du graphe. Tous les arcs dont la source ou le but sont ce nœud sont effacés aussi. $O(d)$
`add_edge(source:vid,target:vid,edge:eid)`	Ajoute un arc d'identifiant `edge`, de source `source` et de but `target` au graphe. $O(1)$
`add_edge(source:vid,target:vid) -> eid`	Identique à la méthode précédente mais génère un identifiant d'arc valide. $O(1)$
`remove_edge(edge:eid)`	Enlève l'arc d'identifiant `edge` du graphe. $O(1)$

TAB. A.4: **Concept** `MutableGraph`**.**

Élément	**Description**
`copy() -> Graph`	Renvoie une copie du graphe courant.

TAB. A.5: Concept `CopyGraph`.

Annexe B

Article : A protocol to analyse cellular dynamics during plant development

The Plant Journal (2005) **44**, 1045–1053 doi: 10.1111/j.1365-313X.2005.02576.x

TECHNICAL ADVANCE

A protocol to analyse cellular dynamics during plant development

Pierre Barbier de Reuille[1], Isabelle Bohn-Courseau[2], Christophe Godin[1] and Jan Traas[2,*,†]
[1]INRA, UMR AMAP, TA40/PSII, Boulevard de la Lironde, 34398 Montpellier Cedex 5, France, and
[2]INRA, Laboratoire de Biologie Cellulaire, Route de Saint-Cyr 78026 Versailles Cedex, France,

Received 28 June 2005; accepted 9 August 2005.
*For correspondence (fax +33(0)4 7272 8600; e-mail traas@versailles.inra.fr).
†Present address: Laboratoire Reproduction et Développement des Plantes, ENS Lyon, 46 Allée d'Italie, 69364 Lyon, France.

Summary

***In vivo* microscopy generates images that contain complex information on the dynamic behaviour of three-dimensional (3D) objects. As a result, adapted mathematical and computational tools are required to help in their interpretation. Ideally, a complete software chain to study the dynamics of a complex 3D object should include: (i) the acquisition, (ii) the preprocessing and (iii) segmentation of the images, followed by (iv) a reconstruction in time and space and (v) the final quantitative analysis. Here, we have developed such a protocol to study cell dynamics at the shoot apical meristem in Arabidopsis. The protocol uses serial optical sections made with the confocal microscope. It includes specially designed algorithms to automate the identification of cell lineage and to analyse the quantitative behaviour of the meristem surface.**

Keywords: confocal microscopy, image analysis, cell lineage, shoot apical meristem.

Introduction

The spectacular advances in imaging techniques have greatly contributed to our understanding of many cellular and molecular processes. However, with the advent of methods that allow the production of massive amounts of data, the imaging field is now facing new requirements. These do not only concern the quality and resolution of the pictures, but also the quantity of the information to be analysed. This is especially true for techniques involving the observation of living tissues. *In vivo* confocal microscopy imaging allows the analysis of cell shape in space and time in intact tissues (Grandjean *et al*., 2004; Reddy *et al*., 2004; Van den Berg *et al*., 1995). In addition, this technique can be combined with GFP technology, thus allowing the observation of specific cellular compartments or gene activity. These methods have opened up a wide range of new possible applications, but they also pose the problem of image and data analysis. Indeed, a major challenge is now to develop the tools to extract pertinent information on the dynamic spatial behaviour of the components that constitute a tissue. To study the dynamics of complex three-dimensional (3D) structures, various types of algorithms must be used. In principle, a complete procedure should include: (i) the acquisition and (ii) initial preprocessing of the images, (iii) image segmentation, (iv) 3D and four-dimensional (4D) reconstruction and (v) the quantitative analysis. The acquisition of the images involves the generation of sets of serial optical sections. This is accompanied by an initial preprocessing (step ii) required to retain as much information as possible. This can vary from a simple setting of contrast to a non-linear filtering to remove unnecessary background noise. The third step, image segmentation, is required to extract and identify the areas to be analysed. More specifically, this implies a classification of the pixels within an image, with their coordinates, as being part of the background or part of the object. For a structure divided into cells, this would typically imply the identification of the pixels corresponding to cell membranes or nuclei for example. Subsequently, the 3D structure of the object is represented in such a way that specific quantitative spatial and temporal characteristics can be extracted, such as changes in cell size, cell shape, neighbourhoods etc. Most of the existing commercial software offers the possibility to

reconstruct 3D objects from serial sections or to extract quantitative information (e.g. on cell size or shape) from the images. However, these tools are usually not able to generate the full protocol able to cover all five steps mentioned earlier.

In this paper, we show how image processing and data analysis can be combined to build up a software chain for 3D and 4D reconstruction. This includes the design of special algorithms to automate the complex processing of time series. To illustrate the approach, we have chosen the shoot apical meristem (SAM) in *Arabidopsis thaliana*. Shoot apical meristems are small groups of dividing stem cells that initiate all the aerial parts of the plant (Traas and Doonan, 2001). Over the last few years, an important amount of information has been accumulated on this structure and *in vivo* techniques to study SAMs in the confocal microscope are now available. It therefore seemed particularly appropriate to develop a full 4D image analysis protocol for this system. However, the approach can be extended to other structures as well.

Results

To visualize the meristems, we used a protocol previously described by Grandjean *et al.* (2004). This method allows the visualization of living meristems at the cellular level in the confocal microscope. The cells either express GFP and can thus be directly visualized, or are stained first using the fluorescent membrane stain FM4-64 (Molecular Probes Europe, Leiden, the Netherlands), which permits the visualization of cell contours and facilitates the reconstruction of cell arrangements in the meristem.

Step 1 and 2: acquisition of images and initial preprocessing in the microscope

Stacks of serial sections were taken at different time points. As the shoot meristem is a flattened structure, transverse sections allowed more easily a complete overview of the meristem than longitudinal sections. The standard software of the microscope was sufficient to guarantee an optimal acquisition of the fluorescent signals (i.e. to obtain images with the proper contrast, representing a maximal amount of relevant signals avoiding saturation or underexposure).

According to the confocal microscope software, sections should be made ideally every 0.36 μm. This distance is linked to the resolution of the confocal microscope along the focal axis. The resolution is determined using the optical laws governing confocal microscopy and mainly the diffraction pattern (see Webb, 1996). However, this implied that at least 100 sections had to be made of every meristem. Because this required an extensive exposure of the tissue to the laser beam, we also tested the quality of the reconstruction when fewer sections were taken (see also Material and methods). This showed that, up to 1 μm between the sections, the quality was still excellent (i.e. 91% of the surface could be reconstructed). Above this value, the errors rapidly increased. For example, at 1.8 μm only 63% of the meristem surface could be digitized. In particular, at the edge of the meristem, the individual cells were no longer clearly distinguishable. Nevertheless, even at a distance of about 2 μm, the results were acceptable, in particular where the cells at the top of the meristematic dome were concerned. In general, a distance between the sections of 1 μm appeared to be a good compromise.

After acquisition, the images were further preprocessed to reduce background noise. This consisted first of the application of a small Gaussian filter of radius between 1 and 5 pixels to smooth images, in particular to reduce the impact of isolated bright or dark dots. In addition, a threshold was applied, removing all the pixels with a brightness less than an empirically defined constant (we used 10 for images with 256 grey levels). This was followed by a linear contrast optimization to ensure that grey levels ranged from 0 to 255.

Step 3: image segmentation; retrieval of geometry and topology

From the stacks of sections, the geometry and the topology of the cells of the surface of the meristem were extracted. In this context, geometry refers to the cell shapes, whereas topology characterizes the neighbourhood relationship (adjacency) between the cells.

The reconstruction of the surface was done in two phases: (i) a reconstruction of the image of the meristem surface from the stack of images; and (ii) definition of the topology and geometry of the cells in 2D.

Reconstruction of the image. The aim was to produce an image of the meristem seen from the top. Because the meristem is a dome with small bumps, there is no structure that overshadows other parts of the surface. As a consequence, the top view permits visualization of the whole meristem surface. To compute such a view, the parts corresponding to the meristem proper have to be separated from the background in each image of the stack (Figure 1). This was achieved by making the area surrounding the meristem transparent, using a so-called transparency mask (see Supplementary Material), while the image of the meristem proper was kept opaque. The image of the meristem surface was finally obtained by looking at the stack of images combined with their transparency mask from the top (Figure 1). Note that, in Figure 1(c), the masked images used intermediate levels of transparency. We found that this allowed a better rendering of the surface image than with binary transparency without loosing useful information (see Supplementary Material).

To compute the transparency masks, we needed to separate the areas outside from the areas inside the

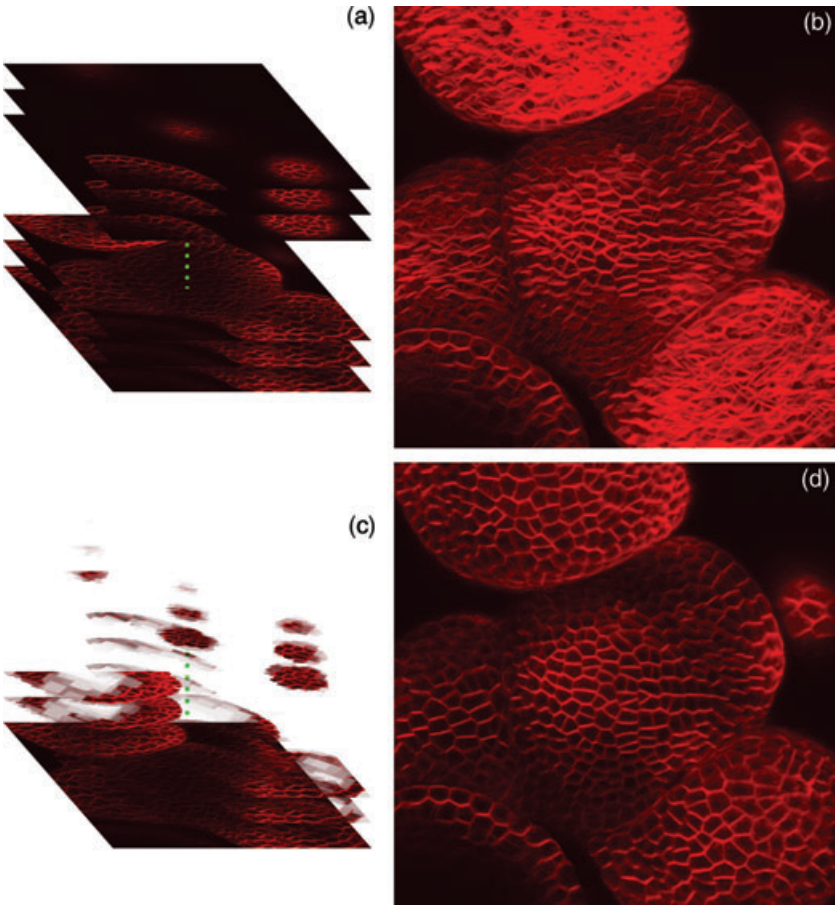


Figure 1. Meristem surface reconstruction: general principle.
The confocal microscope produces a stack of serial sections of the observed meristem (a). The software of the microscope allows a simple projection (b). In this projection, each pixel at position (*x*, *y*) is given the highest value of all pixels at position (*x*, *y*) in the image stack.
The method described in this paper allows the reconstruction of the top view of the meristem surface (d). It first makes the parts outside the meristem transparent, as shown in (c). Then, it constructs a top view of the stack of partly transparent images superposed on a black background. As a result, the surface layer of the meristem can be clearly distinguished.

meristem. The problem was that they both contained black pixels. Figure 2(a) shows a typical image from a stack of sections. In this image, cell walls are very bright while the cell interior and plant exterior are both dark. Fortunately, dark areas within the plants were smaller than the black areas outside the plant. We exploited this structural difference to fill in the smaller dark areas using the topological closure algorithm (see Heijmans *et al.*, 1992; Serra and Vincent, 1992; Vincent, 1992; and Figure 1 in Supplementary Material). This technique fills in gaps having a width smaller than a given threshold. This allowed us to fill in the black pixels corresponding to the interior of the cells. Yet, in our context, this approach had a major drawback: when a separated primordium was too close to the meristem in an image (i.e. closer than the distance threshold, as shown by the white arrow in Figure 2b), the gap between them was filled in as well. To overcome this limitation, we combined the topological closure with a segmentation technique called watersheds (Figure 2).

A grey-scale image (Figure 2a) can be interpreted as a height map (the higher the intensity of the colour, the higher the point is, see Figure 2 in Supplementary Material). In such a 'mountain' map, the interior of every cell is a valley (or watershed) between mountain chains. The dark area outside the meristem forms one or a few very large valleys. A comprehensive definition and an efficient algorithm for watersheds as seen in topology can be found in Vincent and Soille (1991).

After identification of the watersheds in the images (Figure 2c), they were combined with the topological closure. For this purpose, all watersheds covering black pixels in the topological closure image were identified. These were considered as being located outside the meristem and subsequently eliminated (i.e. set to transparent).

As explained above, the images of the stack were superposed with their transparency masks to reconstruct the top view of the meristem.

Definition of two-dimensional geometry and topology. The goal of the reconstruction was to represent the geometry and the topology of the cells in the L1 layer of the meristem. This was achieved with assistance of computer tools specifically developed for this purpose.

Definition of vertices at wall intersection. Cell geometry was most easily defined by identifying manually the points where the walls intersect. Usually, these so-called vertices corresponded to points where three cells were in contact. However,

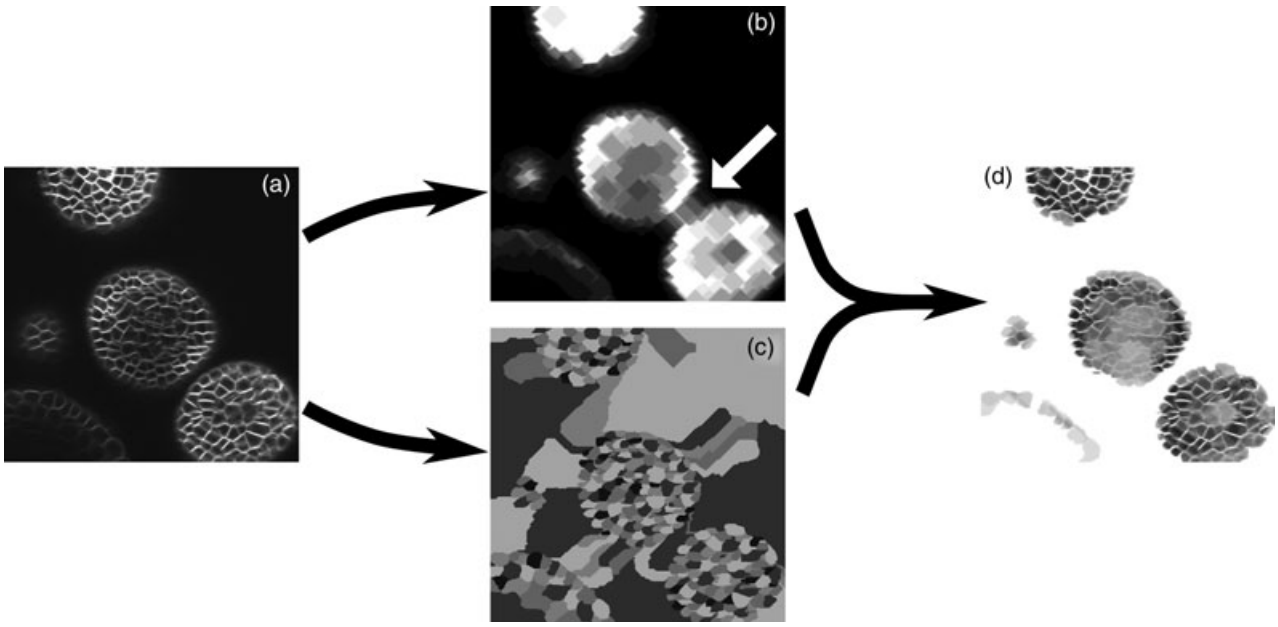


Figure 2. Computation of the transparency mask.
From the original grey-scale image (a) the topological closure is first computed. (b). This technique allows the identification of most of the areas outside the meristem (in black). The major problem with this technique is that it can lead to the apparent connection between the meristem and one of its primordia (white arrow). To correct this, the watersheds are computed. These are shown in (c), where the watersheds are represented as areas with different colours. The area outside the meristem can include several watersheds because of the background noise. The two methods are then combined. Hereby, first all watersheds covering at least one black pixel from the topological closure image are identified. These are considered to be located outside the meristem and entirely set to transparent. The transparency of the other parts is computed according to the topological closure (d).

sometimes, four cells were in contact or two vertices were apparently so close that they could not be distinguished. In this case, we considered that four cells were in contact.

Definition of cell geometry. We chose to represent the geometry of cells by polygons. To define the polygons corresponding to the individual cells, the vertices had to be grouped and ordered. The vertices of individual cells were first grouped manually. To avoid the manual specification of order between vertices, a hypothesis regarding cell shape was made so that the geometry of the cells could be directly inferred from the positions of the vertices. We therefore made the assumption that cell polygons were star-shaped around their centre of gravity. A geometric object is star-shaped around one point p if, for every point p' of the object, the straight segment $[pp']$ is entirely contained within the object. So far, all observed cells could be considered as such.

Definition of cell topology. The groups of vertices corresponding to the cells were then used to compute cell topology. Two cells are neighbours if they have at least one vertex in common. The neighbourhood relationship was represented as a graph (Figure 3, right), where nodes corresponded to

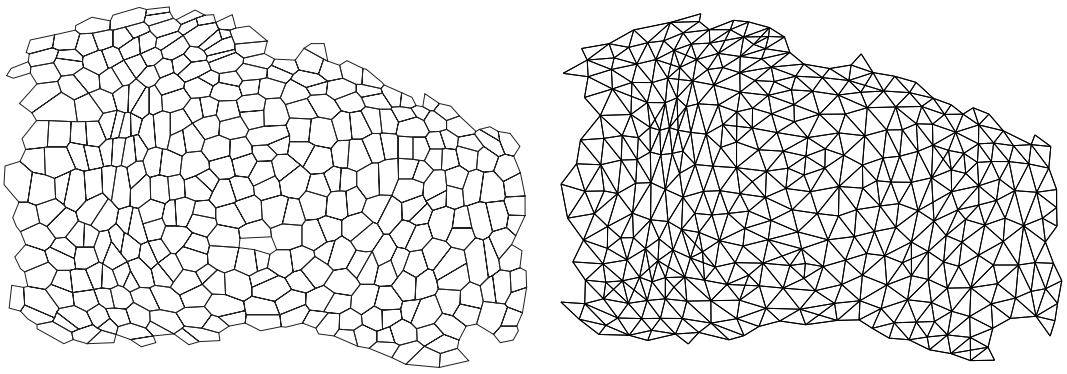


Figure 3. Geometrical and topological representations of the meristem.
The left image represents the reconstructed two-dimensional (2D) meristem with the cell walls (i.e. the geometry). The right image represents the centres of the same cells linked when they share a wall (i.e. the topology).

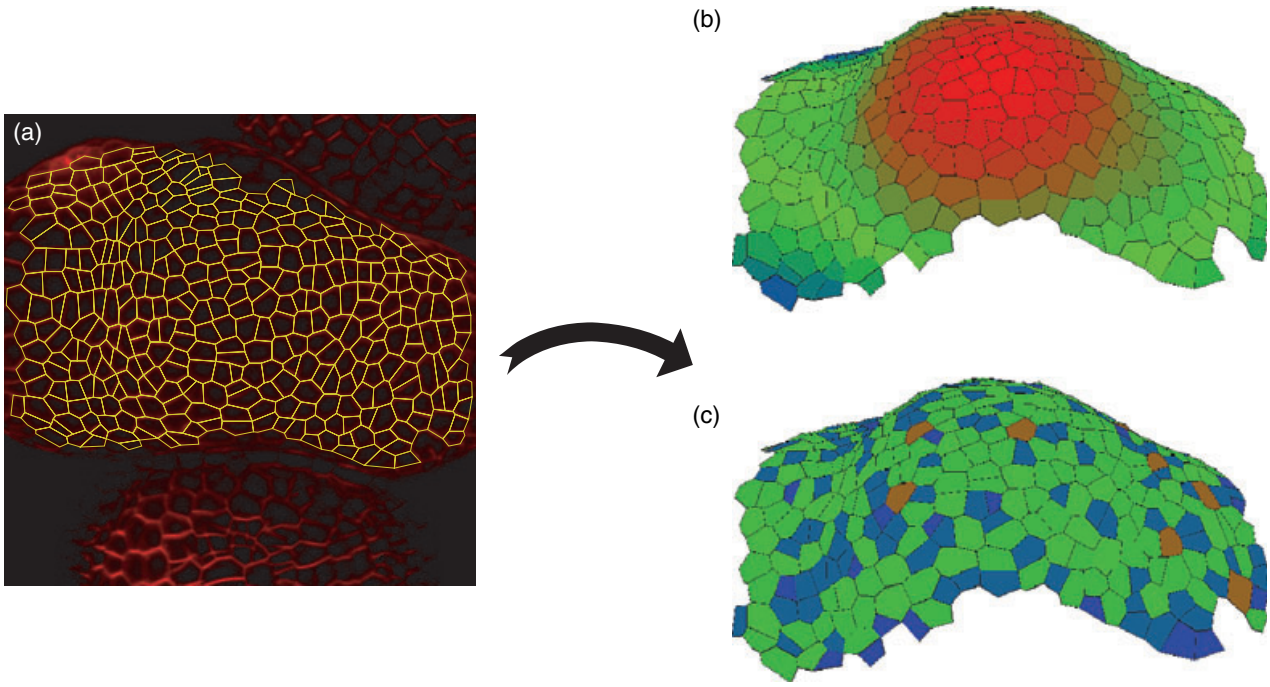


Figure 4. Reconstruction of three-dimensional (3D) surface of the meristem.
The 3D reconstruction of the surface was obtained using the original image stack, the reconstructed top view and the topological and geometrical two-dimensional (2D) reconstruction (a) (for details see text). Once the *z* coordinates are identified, different types of information can be easily extracted, like the *z* coordinates of the cell centres (b) or the number of neighbours of each cell (c). In (b), blue stands for the lowest *z*, red for the highest and green for the medium positions. In (c), cells are colour-coded according to the number of neighbouring cells (dark blue corresponds to four neighbours, blue to five neighbours, green to six neighbours, light green to seven neighbours and brown to eight neighbours).

cells, while the links between nodes represented the adjacency between cells.

Step 4: reconstruction in space (three-dimensional) and time (four-dimensional)

Three-dimensional reconstruction. In step 3, the vertices were identified on a 2D projection (top view) of the dome. In step 4, these vertices were located in the 3D space by identifying them in the original stack of confocal images. Each vertex corresponded to a bright pixel [with coordinates (*x*, *y*)] in the 2D top view. Its *z* coordinate was determined by identifying the section of the stack intersecting the meristem surface with the coordinates (*x*, *y*). This section is the highest one having a pixel at position (*x*, *y*) with a brightness close to that of the 2D projected surface image (Figure 4). The brightness *b* of a point in the stack of images was considered close to the brightness *b*′ of a point on the reconstructed projection if $b > \alpha b'$, where α is an experimental constant in]0,1] (in the current implementation $\alpha = 0.75$ gave excellent results).

Cell lineage identification. To analyse the dynamical behaviour of the tissue, meristems were observed at different time points and the confocal data were used to reconstruct their surface in 3D. As outlined above, these 3D reconstructions were defined as graphs representing both cell geometry and topology. Cells and daughter cells were initially associated manually in the successive reconstructions. However, because this task was very time consuming, we developed an algorithm that semi-automatically followed the cell lineage in a meristem throughout time.

This program requires the manual association of one cell and one of its vertices in the first reconstruction with the corresponding cell and one of its vertices in the second reconstruction. It then goes from neighbouring cell to neighbouring cell, comparing cell shapes and vertex numbers. The combination of these two criteria allows the identification of most of the cells at successive time points and is able to recognize cell divisions. The principle of the algorithm is given in Experimental procedures and computing details are in online material.

Three-dimensional repositioning. Initially, every meristem was reconstructed in the reference system of the confocal microscope (i.e. in the reference system of the stack of images). As the meristems were taken out of the microscope and reinserted between two observations, we had to correct for artefactual reorientation. For this purpose, we readjusted

the position of the meristem reconstruction for each new observation.

For this, we chose a reference system in which the tip of the meristem did not move and in which there is no global rotation of the meristem. The meristem centre was determined visually, based on morphological criteria, in particular with regard to the position of the primordia. For simplicity, the reference system was the one of the microscope at time t_0 (i.e. the time of the first representation of the meristem). As the experimental conditions and the microscope settings were supposed to remain constant through the different observations of a given meristem, the transformation between the microscope and the meristem reference systems was at most the combination of a rotation and a translation. This transformation was computed using a min-square algorithm (see Supplementary Material for details).

Step 5: quantitative analysis

Because the reconstructions contain all the data regarding the neighbourhood relationship, the absolute position of the vertices and the polygon representing the cells, a range of quantitative data can be easily extracted. Because a comprehensive analysis would be beyond the scope of this technical advance article, only a few examples are given in Figures 4 and 6 where the *z* position, the number of neighbouring cells and the speed fields in single meristems are given.

Discussion

To build up our software chain, we had to design and develop new algorithms. In addition, existing algorithms were used and assembled to be adapted to the specificity of our data (e.g. watersheds and filtering algorithms). As a consequence, a key issue in the system design was to integrate all these algorithms within a common toolkit with flexible user interface. We chose to develop a software platform, based on a script language (PYTHON) for which many extension libraries in image processing and data analysis exist. Dumais and Kwiatkowska (2001) used a similar approach to reconstruct and analyse the surface of the meristem from resin replicas observed in the electron microscope. However, their protocol for surface reconstruction was different from ours as it was based on stereo images of entire meristems and not on serial sections. Therefore, their protocol is not applicable to our data. Haseloff *et al.* (2005) developed

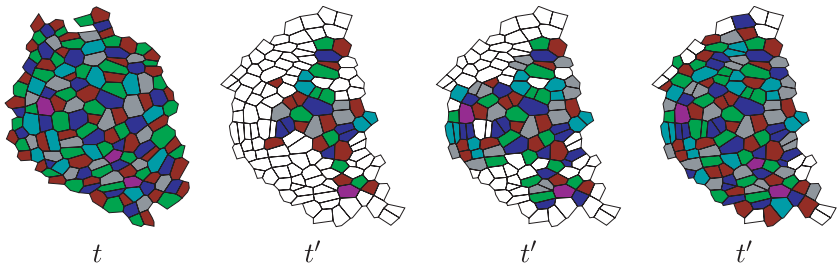


Figure 5. Steps in cell associations.
The figure on the left represents the meristem at time *t*, while the three other figures represent the same meristem at time *t*′. The algorithm identifies identical cells or mother cells and daughter cells at the two time points. The colour is kept constant for a given cell lineage.

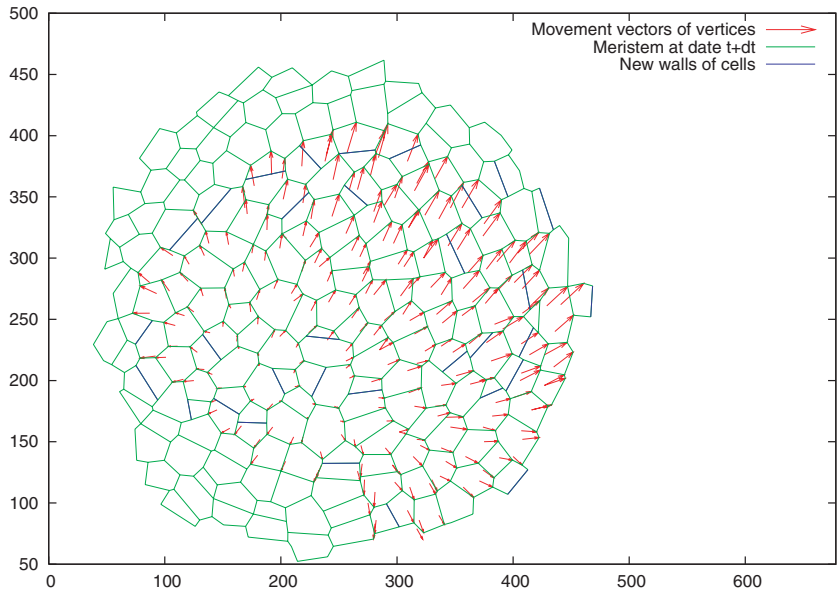


Figure 6. Speed field computed after three-dimensional (3D) repositioning.
Once the successive reconstructions of a meristem are in the same reference system, it becomes possible to compute the movement of the vertices in this reference system.

a software chain for the reconstruction of embryonic tissues and root meristems based on an existing image processing software. Their protocol allows for the reconstruction of geometry and topology of cells in intact tissues. However, their protocol requires a very high definition of the images, an optimal amount of sections and specific stains to differentiate different parts of the cells (Haseloff, 2003). Therefore, fixed material and post-fixation treatments have to be used to improve the final image quality. The images obtained from the living meristems using our method clearly did not have the same resolution. This is not only because the staining method results in more background noise, but also because the number of sections through the living meristems has to be kept to a minimum. Therefore, we developed a different approach. It is true that our method is comparatively time consuming. However, this disadvantage is compensated by an increased robustness.

An aspect that has not been considered in this study is the 3D rendering of internal parts of the meristem. Even if the 3D geometry of cells can be approximated by 3D polyhedrons, it is difficult to define the sides and vertices of the internal cells visually and manually. With the aim of developing an automatic (or semi-automatic) algorithm, we made some preliminary studies on a completely automatic way to detect cells and extract their geometry, but this turned out to be a very difficult task. The method proposed by Haseloff *et al.* (2005) allows a semi-automatic reconstruction of the geometry and topology of the cells. However, as pointed out above, the reconstruction algorithm works on high-resolution images of fixed material, which is, in principle, not applicable to our data. By any means, this type of semi-automatic approach requires the manual identification of the cells, which is a time consuming task, in particular if time series of individual meristems are to be studied. Therefore, we conclude that, at this stage, a comprehensive analysis of cell shape dynamics in the internal parts of the meristem would require substantially more work.

Sources of error

Once the 4D reconstructions have been obtained, it can be difficult to recognize, *a posteriori*, artefacts that have occurred during the procedure and that can influence the final analysis. It is therefore important to identify such potential sources of errors at the moment they occur.

The first source of error is due to the confocal microscope itself. In first approximation, the obtained image of the meristem is not the meristem itself but the meristem convoluted by a function (called the point spread function). This function is governed by wave optics and is the cause of the resolution limits of the microscope (Webb, 1996). These limits, in turn, cause a certain imprecision in the positioning of the vertices. The imprecision is greater along the *z*-axis (i.e. along the focal axis) compared to the *x* and *y* axes (i.e. in the focal plane). The error in the focal plane can lead to an apparent merging of vertices that are very close to each other. This subsequently results in the generation of reconstructions where certain vertices are formed by four different cross-walls. However, this only concerns a limited number of cells and has not been a major source of artefacts. In more general terms, the errors generated by the resolution limits are well under the typical size of a cell and do not pose a major problem in the final quantitative analysis. A potentially important source of error is a shape distortion of the objects due to optical aberrations or problems with the scanning device. Fortunately, these can easily be identified by using fluorescent beads with known size. A second problem along the *z*-axis is related to the thickness of the sections and the distance between them, which can lead to a mislocalization of the vertices in the final reconstruction. In the case of one optical section every 2 μm (approximately), this error can be up to 1 μm at worst.

Conclusion

Different methods are currently available to analyse the dynamics of a 3D structure. Although the general approach is always the same, there does not seem to be a single standardized method available that is immediately applicable to all objects. As a consequence, the existing algorithms have to be adapted to the biological system that is used. Here, we have described a full protocol to analyse the growth and development of the SAM. The method is robust and gives access to a very wide range of quantitative data. In more general terms, the type of analysis described here will undoubtedly become more and more important in the future.

Experimental procedures

Plant culture and confocal microscopy

The plants were grown and observed essentially as described by Grandjean *et al.* (2004). Seeds were sown on petri dishes with a medium adapted for Arabidopsis (Hamant *et al.*, 2002). After 2 days at 4°C, the seeds on medium were put in growth chambers at 20°C and 16 h of light. For napthyphthalamic acid (NPA) treatment, 10^{-5} to 10^{-6} M of NPA was added to the medium. As soon as naked inflorescences had formed, the plants were transferred to medium without the inhibitor. As soon as these inflorescences started to generate new flower buds, they were examined with a Leica TCS-NT confocal laser scanning microscope (Leica, Heidelberg, Germany) with an argon/krypton laser (Omnichrome, Chino, CA, USA) and a AOTF (Acouster-Optical Tunable Filter) for excitation. For this purpose, the meristems were embedded in a layer of low melting point agarose (Sigma, St Louis, MO, USA), the tip of the meristem being close to the bottom of a WillCo-dish (WillCo Wells, Amsterdam, the Netherlands) with a glass bottom. Best results were obtained when the meristems were embedded while the agarose was not yet completely solid. The red vital dye FM4-64 was used to visualize the cells. Medium scan (450 lines per second) images (512 × 512 pixels) were generated using a long-distance Leica 40 × 0.8 NA water HCX APO L (Leica). The

inflorescences were observed in an inverted microscope DM IRB (Leica, Germany). As a consequence, the plants were observed with their meristem down. As soon as a stack of images was obtained on a particular meristem, the plant was put back in the growth chamber, meristem up, to recover.

Quantification of errors

Microscopy errors. Potential optical aberrations due to problems with the lenses or with the scanning devices of the microscope can lead to important imprecision in the quantification of cell size and shape. Because they cannot be evaluated on a theoretical basis, they have to be estimated experimentally. For this purpose, we replaced the meristems by calibrated fluorescent spheres with a diameter 84 μm. To distinguish between the errors due to the lens and other factors, we also measured the spheres with another lens. This showed that our equipment did not lead to significant distortions.

Digitizing errors. To evaluate the quality of the suboptimal reconstructions, we compared them to the optimal reconstruction using three different criteria: (i) the surface of the meristem covered by the representation; (ii) the number of missed and added ('phantom') vertices when compared with the optimal reconstruction; and (iii) the calculated mean distance between corresponding vertices in the optimal and the suboptimal reconstructions. The values computed for these criteria are given in Figure 7.

Implementation

Only a very general description of the protocol and algorithms will be given here. The process implied two different pieces of software developed especially for this purpose on GNU/Linux. The first one is called MERRYPROJ and is used to compute the projection of the surface of the meristem. The second one is called MERRYSIM and is used to reconstruct the 3D-meristem from the projection and perform further analysis.

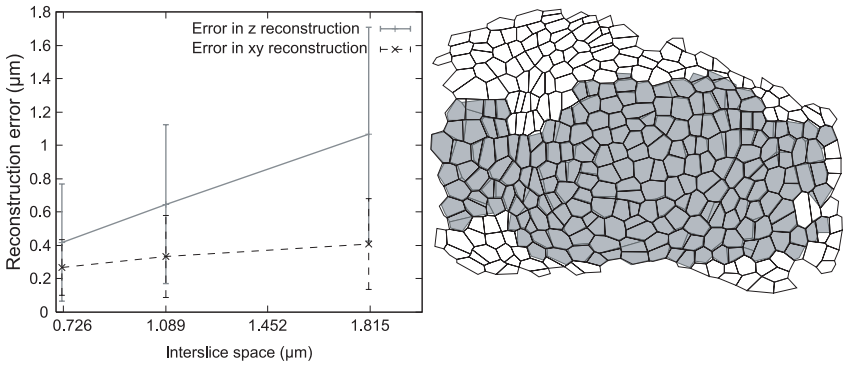
The user interface of MERRYPROJ was created using PYTHON and the PyQt graphical toolkit, but the algorithms were implemented in C++ and used to extend PYTHON using the Boost.Python library (http://www.boost.org/libs/python/index.html).

MERRYSIM was mainly created using C++ and Qt and embeds a PYTHON interpreter to perform analyses. Both pieces of software are distributed under the terms of the General Public Licence (http://www.gnu.org/licenses/gpl.html) and are freely available at ftp://ftp.cirad.fr/pub/amap/VirtualPlants/merrysim/. Moreover, all the details needed to implement the algorithms are described in the Supplementary Material.

Cell lineage algorithm

The algorithm starts from a single cell and tries to identify its neighbours in two consecutive reconstructions of a meristem taken at time points t and t' ($t' > t$) as presented in Figure 8. At the each step, the algorithm needs the association of one cell C and one of its vertices at time t, with the corresponding cell C' and one of its vertices at time t' (at the first step it is done manually). In case the cell C divided between t and t', C' corresponds to the merged daughter cells of C (see Supplementary Material). This implies that the new vertices formed during division of C are ignored until the end of this step.

The algorithm next associates every vertex of C with the corresponding vertex of C'. This is achieved in the following manner. If there was no cell division of the neighbouring cells, the mapping of the vertices of C on the vertices of C' is straightforward. Otherwise, the algorithm localizes the vertices in C' resulting from the divisions. This is illustrated in Figure 8. In this figure, there is one more vertex in C' than in C. To find out which vertex is new in C', all the polygons created from C' by removing one vertex are compared to C. The orange polygon in the figure is the one closest to C and the algorithm concludes that v'_7 is the new vertex. To associate the vertices of C with the ones of C', the algorithm starts from v_1, which is known to be associated with v'_1 and follows C and the orange polygon in the same way, associating the vertices $v_{2\ldots6}$ with the vertices $v'_{2\ldots6}$ excluding v'_7. The neighbours of C are then



Inter-slicesspace	Surface coverage	Missed vertices	Phantom vertices
0.726 μm	93.6%	75	18
1.089 μm	91.6%	85	50
1.815 μm	63.6%	283	19

Figure 7. Optimal amount of sections for reconstruction.
The top left graphic represents the reconstruction error (i.e. the distance between the vertices in the suboptimal reconstruction and the optimal one) in the z coordinate and in the (x, y) coordinates. These two kinds of error were separated because they come from different reconstruction steps. The error made seems to be a linear function of the distance between the sections. The graphic shows the mean values (linked by a line) and the standard deviations.
The top right image represents an optimal reconstruction (white) and a reconstruction made with sections at a distance of 1.815 μm (in grey). Note that, even at 1.815 μm, an important part of the meristem can be studied.
The table presents quantification of structural errors (i.e. missing or added vertices in the suboptimal reconstructions) and the surface coverage for each tested intersection space. Note that the optimal reconstruction includes 722 vertices.

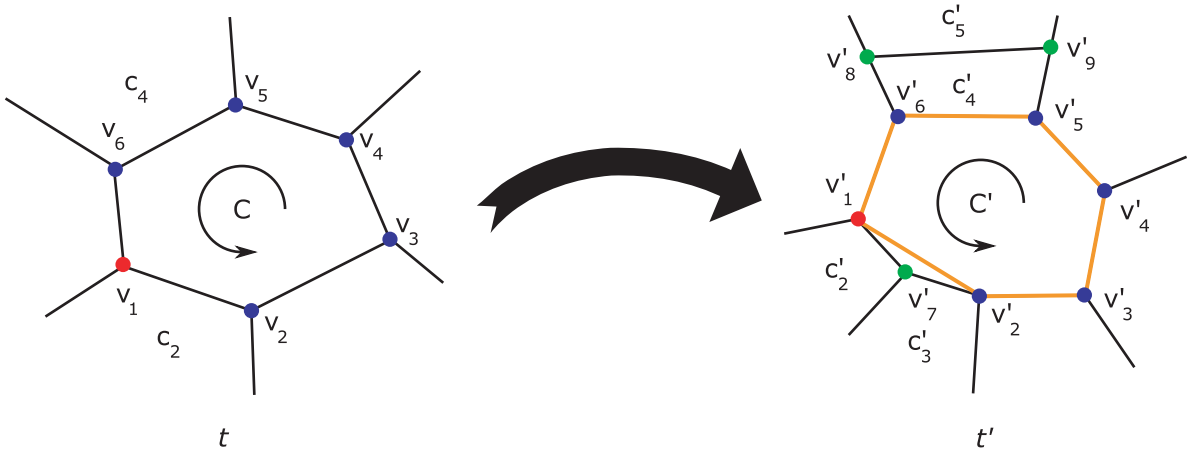


Figure 8. Identification of neighbouring cells.
C and *C′* correspond to the same cell in two successive representations of the same meristem at time points *t* and *t′*. v_1 and v'_1 correspond to the same vertex at the two time points. From this knowledge, the algorithm tries to associate the neighbours of *C* with those of *C′*. The association of the vertices of *C* with those of *C′* is enough to associate the neighbours so that each neighbour c_n of C is associated with a neighbour c'_m of *C′* if and only if c'_m is either the same cell as c_n or a daughter-cell of c_n. In this figure, there is one more vertex in *C′* than in *C*. To find out which vertex is new in *C′*, all the polygons created from *C′* by removing one vertex are compared to *C*. The orange polygon in the figure is the one closest to *C* and the algorithm concludes that v'_7 is the new vertex.

associated with the neighbours of *C′* using the association of the vertices (see Supplementary Material). This then allows the algorithm to map the vertices of *C* onto the corresponding ones of *C′*.

The mapping of the vertices directly allows the association of the edges of the polygons representing *C* and *C′*. Because every edge also belongs to one and only one polygon of the neighbouring cell, it is possible to associate the neighbouring cells of *C* with the neighbouring cells of *C′*. For each neighbour of *C*, three cases can be distinguished: (i) the neighbouring cell has not divided; (ii) the neighbouring cell has divided such that a new vertex has formed in *C′* (see Figure 8 where c_2 has divided into c'_2 and c'_3); and (iii) the neighbouring cell has divided but no new vertex has formed in *C′* (Figure 8 where c_4 has divided into c'_4 and c'_5).

In the first case, the algorithm simply associates the neighbouring cell of *C* with the one of *C′*. In the second case, the algorithm detects the division thanks to the presence of the new vertex. As a result, the two daughter cells at *t*0 are associated with their mother cell at *t* (c_2 in Figure 8). However, in case (iii) no new vertex has formed and, in our example, only c'_4 will be associated with c_4 and not c'_5. As a consequence, in contrast to the two other cases, this association is not complete as one of the daughter cell is missing. Because based only on the association of local vertices the algorithm cannot distinguish between complete and incomplete cell associations, they all need further processing. To determine which ones are complete, the algorithm uses a heuristic: a confidence score is computed for each cell association, based on a comparison of their shapes (see Supplementary Material), which will be very different if a mother cell is compared to only one of its daughter cells. The associations and their confidence scores are subsequently added to a global list containing all undetermined associations. The heuristic then considers the association of the list with the highest confidence score as complete. It is removed from the list and the algorithm starts the whole procedure again from the selected associated cells, until the list of undetermined associations is empty.

Some steps of the association process are shown in Figure 5.

Supplementary Material

The following supplementary material is available for this article online:

Appendix S1.
This section contains a detailed description of the algorithms used.

References

Dumais, J. and Kwiattowska, D. (2001) Analysis of surface growth in shoot apices. *The Plant Journal*, **31**, 229–241.

Grandjean, O., Vernoux, T., Laufs, P., Belcram, K., Mizukami, Y. and Traas, J. (2004) In vivo analysis of cell division, cell growth, and differentiation at the shoot apical meristem in arabidopsis. *Plant Cell*, **16**, 74–87.

Hamant, O., Nogue, F., Belles-Boix, E., Jublot, D., Grandjean, O., Traas, J. and Pautot, V. (2002) The knat2 homeodomain protein interacts with ethylene and cytokinin signaling. *Plant Physiol.* **130**, 657–665.

Haseloff, J. (2003) Old botanical techniques for new microscopes. *Biotechniques*, **34**, 1174–1182.

Haseloff, *et al.* (2005) Website: http://www.plantsci.cam.ac.uk/Haseloff/Home.html

Heijmans, H., Nacken, P., Toet, A. and Luc, V. (1992) Graph morphology. *J. Vis. Commun. Image Representation*, **3**, 24–38.

Reddy, G.V., Heisler, M.G., Ehrhardt, D.W. and Meyerowitz, E.M. (2004) Real-time lineage analysis reveals oriented cell divisions associated with morphogenesis at the shoot apex of arabidopsis thaliana. *Development*, **131**, 4225–4237.

Serra, J. and Vincent, L. (1992) An overview of morphological filtering. *Circuits Systems Signal Process.* **11**, 47–108.

Traas, J. and Doonan, J. (2001) Cellular basis of shoot apical meristem function. *Int. Rev. Cytol.* **208**, 161–206.

Van den Berg, C., Willemsen, V., Hage, W., Weisbeek, P. and Scheres, B. (1995) Cell fate in the arabidopsis root meristem determined by directional signalling. *Nature*, **378**, 62–65.

Vincent, L. (E. Dougherty ed.) (1992) Morphological algorithms. In: *Mathematical Morphology in Image Processing*. Marcel-Dekker, pp. 255–288.

Vincent, L. and Soille, P. (1991) Watersheds in digital spaces: An efficient algorithm based on immersion simulations. *IEEE Trans. Pattern Anal. Mach. Intell.* **13**, 583–598.

Webb, R.H. (1996) Confocal optical microscopy. *Rep. Prog. Phys.* **59**, 427–471.

Annexe C

Article : Computer simulations reveal properties of the cell-cell signaling network at the shoot apex in *Arabidopsis*

Computer simulations reveal properties of the cell–cell signaling network at the shoot apex in *Arabidopsis*

Pierre Barbier de Reuille*†, Isabelle Bohn-Courseau†‡, Karin Ljung§, Halima Morin‡, Nicola Carraro¶, Christophe Godin‖, and Jan Traas‡††**

*Institut National de la Recherche Agronomique and ‖Institut National de Recherche en Informatique et en Automatique, Unité Mixte de Recherche Botanique et Bio-informatique de l'Architecture des Plantes, TA40/PSII Bd de la Lironde, 34398 Montpellier Cedex 5, France; ‡Laboratoire de Biologie Cellulaire, Institut National de la Recherche Agronomique, Route de Saint-Cyr, 78026 Versailles Cedex, France; §Department of Forest Genetics and Plant Physiology, Umeå Plant Science Centre, Swedish University of Agricultural Sciences, S-901 83 Umeå, Sweden; and ¶Università Degli Studi Di Padova, Agripolis, Legnaro 35020, Italy

Communicated by Elliot M. Meyerowitz, California Institute of Technology, Pasadena, CA, November 23, 2005 (received for review August 10, 2005)

The active transport of the plant hormone auxin plays a major role in the initiation of organs at the shoot apex. Polar localized membrane proteins of the PIN1 family facilitate this transport, and recent observations suggest that auxin maxima created by these proteins are at the basis of organ initiation. This hypothesis is based on the visual, qualitative characterization of the complex distribution patterns of the PIN1 protein in *Arabidopsis*. To take these analyses further, we investigated the properties of the patterns using computational modeling. The simulations reveal previously undescribed properties of PIN1 distribution. In particular, they suggest an important role for the meristem summit in the distribution of auxin. We confirm these predictions by further experimentation and propose a detailed model for the dynamics of auxin fluxes at the shoot apex.

auxin | modeling | shoot meristem

There is strong evidence that active auxin transport, generated by influx and efflux carriers, creates patterns of auxin distribution at the shoot apex. This distribution is, in turn, interpreted in terms of differential growth and cell differentiation (1–3). In *Arabidopsis*, AUX1, a putative influx transporter (4), is mainly located in the surface layer (L1) of the shoot apical meristem (2) (Fig. 1*A*). Interestingly, the protein seems to be homogeneously distributed in plasma membranes of the individual cells. Therefore, it has been proposed that AUX1 helps to restrict auxin to these layers, although additional mechanisms may be required (5). The efflux facilitator PIN1 also is localized in the surface layers of the meristem, but in contrast to AUX1 it is often localized on certain anticlinal sides of the cells only. Because neighboring cells often show coherent PIN1 positioning, it was proposed that PIN1 is responsible for directed hormone flows within the meristem L1 layer (Fig. 1*A*). In particular, careful immunological studies have revealed that the membranes carrying PIN1 are preferentially oriented toward the incipient primordia, suggesting auxin transport toward the young organs (2, 3).

Together, the observations so far suggest a dynamic scenario where auxin is transported to the meristem from basally localized tissues via the L1 layer. At the meristem surface, auxin is redistributed and accumulates at particular sites where it will induce the initiation of new organs. This accumulation subsequently leads to the activation of transport in the provascular tissues causing an inward directed flow (Fig. 1*B*). The young organ is thus transformed into an auxin sink, which depletes its surroundings from auxin and prevents the formation of new primordia in its vicinity.

Although this scenario is relatively straightforward, the previous observations leave a number of questions open. First, it is not clear at all why auxin should start to accumulate at the site where a primordium will be initiated. Second, the immunolabelings reveal a very complex distribution of PIN1 proteins (Fig. 2). As a result, the interpretation of these patterns in terms of cell–cell interaction networks and, more specifically, in terms of auxin distribution remains extremely difficult.

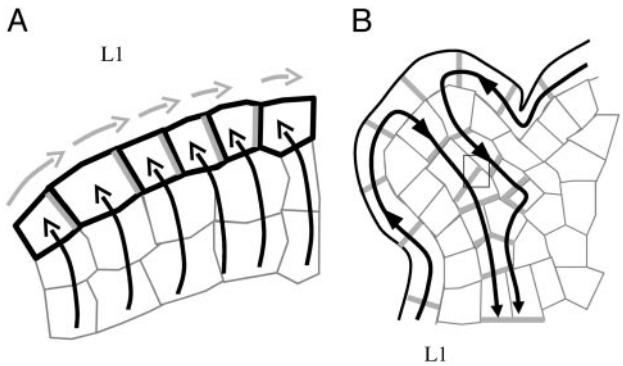


Fig. 1. Models for auxin transport in the shoot apical meristem. (*A*) The putative auxin influx carrier AUX1, represented in black, is homogeneously distributed on the cell membranes of the surface layer of the meristem, whereas the putative auxin efflux carrier PIN1, represented in gray, seems to have a polarized localization. As proposed by ref. 2, AUX1 would help to concentrate auxin in the surface layer (black arrows), and PIN1 would direct auxin fluxes (gray arrows) within these layers. Note that additional mechanisms responsible for auxin influx into the L1 layer have been proposed (5). (*B*) In the provascular tissues of young primordia, PIN1 is oriented downward, evacuating auxin from the meristem surface (black arrows) to deeper tissues. Consequently, the primordia act as auxin sinks.

To address these questions, we developed computational modeling tools that allowed us to uncover previously undescribed properties of the cell–cell interaction network and to predict auxin fluxes in the shoot apical meristems directly based on microscopical observations.

Results

Sections of shoot apical meristem were labeled with anti-PIN antibody (Ab). To interpret the complex labeling patterns in term of putative auxin distribution, we simulated the hormone

Conflict of interest statement: No conflicts declared.

Abbreviation: GCMS, gas chromatography and MS.

†P.B.d.R. and I.B.-C. contributed equally to this work.

**To whom correspondence may be addressed. E-mail: godin@cirad.fr.

††To whom correspondence may be sent at the present address: Laboratoire Reproduction et Développement des Plantes, Ecole Normale Supérieure Lyon, 46 Allée d'Italie, 69364 Lyon, France. E-mail: jan.traas@ens-lyon.fr.

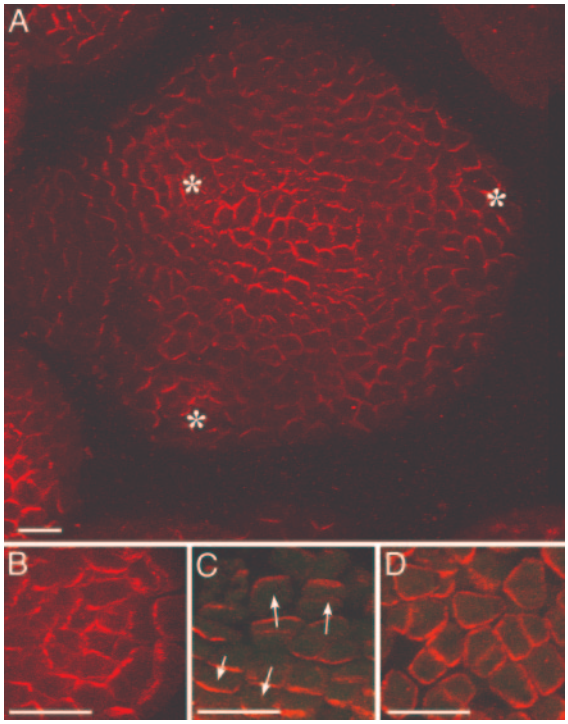


Fig. 2. PIN1 immunolocalization in *Arabidopsis* shoot apical meristems (6). (*A*) Global view of an anti-PIN1 immunolabeling on a meristem cross section. PIN1 is localized on the membrane and polarized in most cells. Patterns are complex. Asterisks, young primordia. (Bar, 20 μm.) (*B*) In the peripheral zone of the meristem, concentric PIN1 orientations around young primordia are observed. The patterns suggest that the cells orient toward a single central cell of the primordium. (*C*) In boundaries between the meristem and the primordium, cell polarities in opposing directions are observed (arrows). (*D*) At the meristem summit, PIN1 localization is variable and does not seem to show any particular organization. (Scale bars for *B*–*D*, 10 μm.)

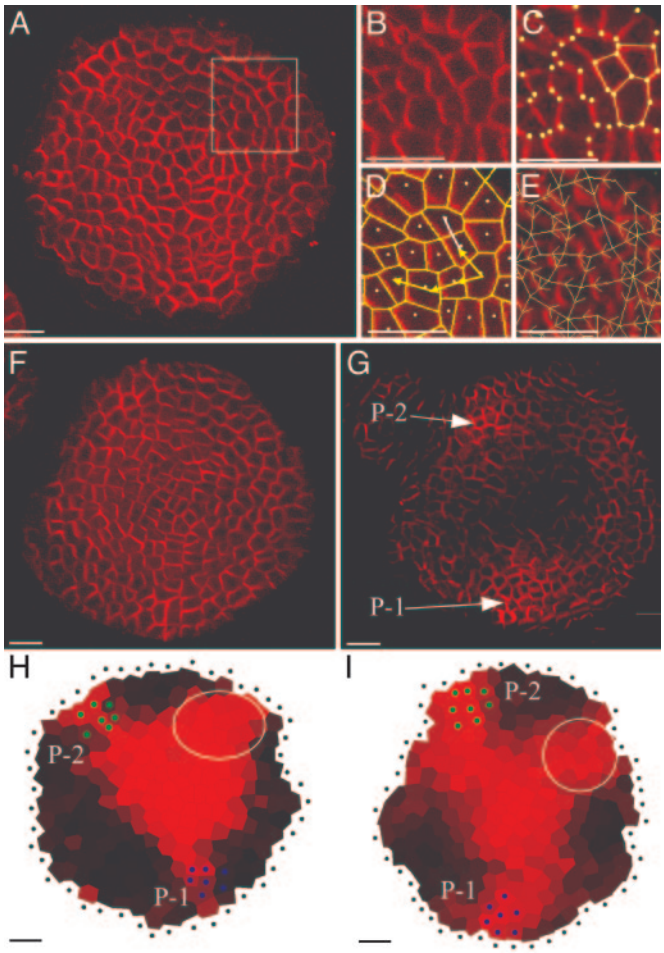


Fig. 3. From PIN1 immunolabeling to the simulation of auxin fluxes. (*A*) A transverse section showing PIN labeling. [Reprinted with permission from ref. 7 (Copyright 2005, Elsevier).] The rectangle indicates the detail shown in *B*. MERRYSIM (see *Supporting Text*) is used to capture the cell shapes and the PIN1 localization in each cell. (*C*) All cell vertices (spots) are manually positioned. The vertices of each cell are subsequently grouped. (*D*) Cells are manually connected to each other if and only if there is a PIN1 labeling on the membrane between them (arrows). The connection is oriented in the way of supposed PIN1-mediated efflux. (*E*) The result is a network of cell interactions. (*F* and *G*) Anti-PIN1 immunolabeling on two successive transverse sections of another meristem. In *G*, the labeling of the provascular strands at the level of P1 and P2 can be clearly distinguished (arrows). At these positions, called the primordium centers, auxin will be evacuated in the simulations. (*H* and *I*) Results of the simulated auxin fluxes in meristems shown in *A* and *F*. The position of the primordium centers visible on the original images are marked by green and blue dots. Virtual auxin is injected via the black dots surrounding the meristems. The quantity of virtual auxin per cell is proportional to the red intensity. Auxin accumulates where young primordia are being formed, but also at the meristem summit. Moreover, the auxin maximum at the meristem summit protrudes toward the initium I-1 (gray circle). (Scale bars, 20 μm.)

fluxes on digitized meristems (Figs. 3 *A*–*G* and 4 *A* and *B*; also see *Materials and Methods*).

Simulation of Auxin Fluxes. The auxin transport through the network of interconnected cells was modeled by using the following set of hypotheses:

(*i*) Auxin passively diffuses via all walls (edges of the individual cells in the graph) and is actively transported via oriented connections only (1–4, 8–11). We only consider net auxin flux from cell to cell, without taking into account the molecular mechanisms involved (5, 12–14). To keep a tractable model at the tissue level, we decided to model this transport process using a simplified system, where we do not represent the compartment corresponding to the intercellular space. The net balance of auxin in a cell thus is considered to be the result of a direct exchange between cells through two processes: diffusion from cell to cell and polarized active transport due to the presence of PIN1 molecules on certain membranes of cells.

(*ii*) Auxin is restricted to the L1 layer and enters the meristem from the meristem border via the efflux facilitator (2) or, alternatively, auxin produced by every cell within the meristem.

(*iii*) Auxin is evacuated via the L1 cells that are in contact with provascular strands characterized by PIN1 labeling in deeper layers (1, 2) (Fig. 3*G*). Longitudinal sections show that these provascular strands are approximately three cells wide (data not shown). Therefore, a circular area of three cells wide is designated to evacuate auxin at the position of each provascular strand on the images. They are defined here as "Primordia" (P-1, P-2, . . . , P-1 being the nearest to the meristem summit) and behave as auxin sinks.

(*iv*) The simulation algorithm continues to distribute the virtual auxin in the system until the auxin distribution gets stationary. This hypothesis is to take into account that the establishment of auxin distribution is a fast process, much faster than growth and cell proliferation (2). Therefore, in a normally growing meristem, auxin distribution is likely to be near the equilibrium at all times.

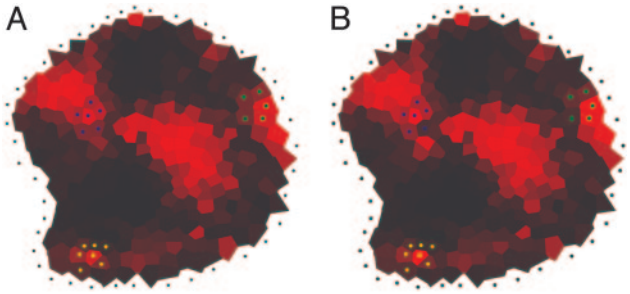


Fig. 4. Effect of stronger active transport in primordia. (*A*) Reference simulation with uniform active transport. (*B*) Simulation with a 10-fold increase of active transport in the cells of the different primordia (colored dots).

(*v*) Cells cannot accumulate auxin indefinitely. We modeled this constraint by using a saturation level, above which the cells no longer accept auxin influx. Simulations tests showed that this scenario was not very different from the situation where, at high level of accumulation, auxin diffusion overcomes active transport (see *Supporting Text*, which is published as supporting information on the PNAS web site).

(*vi*) Auxin is degraded at a constant rate in each cell. Situations with different degradation levels were tested, including no degradation at all.

Ten meristems that were precisely sectioned in a plane transverse to the stem were immunolabeled (6) by using the PIN Ab. Subsequently, the corresponding images were used to extract connection maps. When the six rules mentioned above were applied to these maps (for technical details, see *Supporting Text* and Figs. 9–14 and Tables 1 and 2, which are published as supporting information on the PNAS web site), virtual auxin accumulated at the sites where young primordia were being formed (Fig. 3 *H* and *I*). This property of the PIN1 network could be expected from visual inspection of the immunolabelings. However, the simulations also showed a strong accumulation of virtual auxin in a domain covering the meristem summit, a property not obvious from visual inspection only. In all meristems tested, the central zone of accumulation also locally extended further to the periphery. Interestingly, this peripheral protrusion corresponded precisely to the site where the organ founder cells of the next primordium (called here initium-1 or I-1) were expected. More specifically, the divergence angle between the last formed primordium and this initium oscillated between 105° and 145° with a mean at 130°. Extensive tests showed that the patterns were robust, relatively insensitive to even major changes in the parameters (see Fig. 4 and *Supporting Text*).

Auxin at the Meristem Summit. An unexpected simulation result was that the meristem summit accumulated auxin, suggesting a role for this domain in hormone distribution. Because previous studies only indicated a minor role for the meristem summit in this respect (15), we next tested this prediction in planta. We first analyzed plants expressing *GFP* under control of *pDR5*, a synthetic promoter that is sensitive to auxin and that has been used to estimate relative hormone threshold levels in different tissues (3). As expected, GFP was strongly expressed in the future organ primordia, even at very early stages of initiation, i.e., at the level of I-1, just next to the meristem summit (Fig. 5*A*) (3). As a consequence, this pattern fully coincided with those predicted by the simulations. However, in contrast to what could be expected from the simulations, GFP was not, or very weakly, expressed in the meristem summit. Therefore, either this domain contained little or no auxin, or *pDR5* was insensitive to auxin in the meristem summit. To distinguish between these two possibilities, we treated young *in vitro* grown plants (18) expressing *pDR5*::*GFP* with auxin in absence or presence of the auxin transport inhibitor NPA. The presence of 10^{-5} M auxin and 10^{-5} M NPA caused an important increase in the amount of *pDR5*::*GFP* expressing cells. However, the meristem summit never showed any increase in GFP activity, even in meristems where the entire periphery had activated the marker (Fig. 5 *B–E*). We concluded that, as judged by *pDR5* activity, the central domain of the meristem was auxin-insensitive. The observed insensitivity did not provide any information on the actual amount of auxin present in this domain. To address this issue, we used a monoclonal Ab directed against auxin to define local differences in auxin concentrations (19). The labeling showed a weak, but consistent, pattern, with an obvious maximum at the meristem summit (Fig. 5 *F* and *G*).

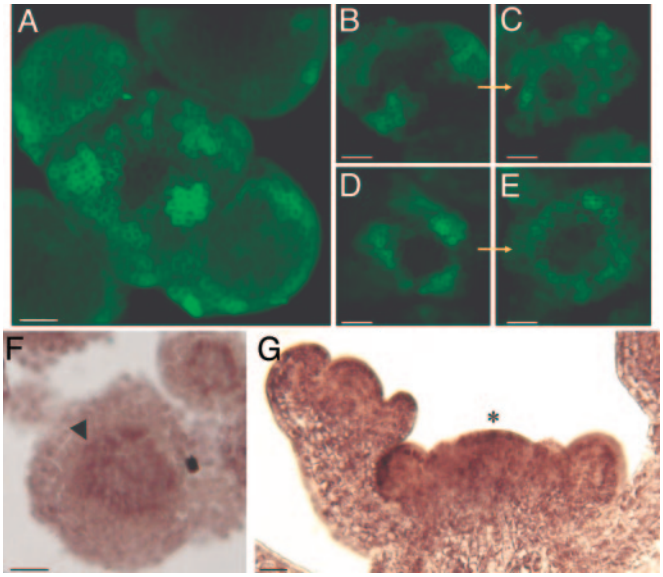


Fig. 5. Localization of auxin in *Arabidopsis* shoot apical meristems. (*A–E*) Spatial pattern of *pDR5*::*GFP* expression in shoot apical meristems under different conditions. (*A*) Untreated meristem. (*B* and *C*) Treatment of a meristem with 10^{-5} M IAA during 22 h. 10^{-5} M NPA (auxin transport inhibitor) was added to keep auxin in the meristem (*B*, $t = 0$ h; *C*, $t = 22$ h). (*D* and *E*) Treatment of a meristem with 10^{-5} M of the synthetic auxin 2,4 D during 22 h (*D*, $t = 0$ h; *E*, $t = 22$ h). The *pDR5*::*GFP*-expressing domain covers a larger part of the periphery after the treatment with IAA-NPA or 2,4 D but the summit of the meristem remains unlabeled.(*F* and *G*) Immunolocalization of IAA in shoot apical meristems (16, 17). The presence of labeling is characterized by a purple/brown signal. (*F*) Cross section of a wild-type meristem; showing labeling at the meristem summit (arrowhead). (*G*) Longitudinal section of a wild-type meristem also showing labeling at the meristem summit (asterisk). (Scale bars, 20 μm.)

To provide additional evidence that auxin did accumulate in the central part of the meristem, we extended our analysis to gas chromatography and mass spectrometry (GCMS) (20, 21). Because a normal wild-type meristem was too small to perform this type of analysis, we decided to use the *clavata3* (*clv3*) mutant. This mutant lacks a signaling peptide (CLV3) that is required to keep the central part of the meristem within certain size limits (22, 23). If this peptide is absent, the central domain continues to grow, until it is several millimeters wide (Fig. 6*A*). To confirm that the central domain of the *clv3* meristem behaved like a normal wild-type meristem summit with regard to auxin sensitivity, we crossed the *pDR5*::*GFP* marker into the mutant background. In the enlarged dome of the mutant, we could only observe GFP fluorescence at the very periphery, close to the site of organ initiation (Fig. 6 *B* and *C*). The rest of the enlarged meristem did not express DR5-GFP. This result confirmed that

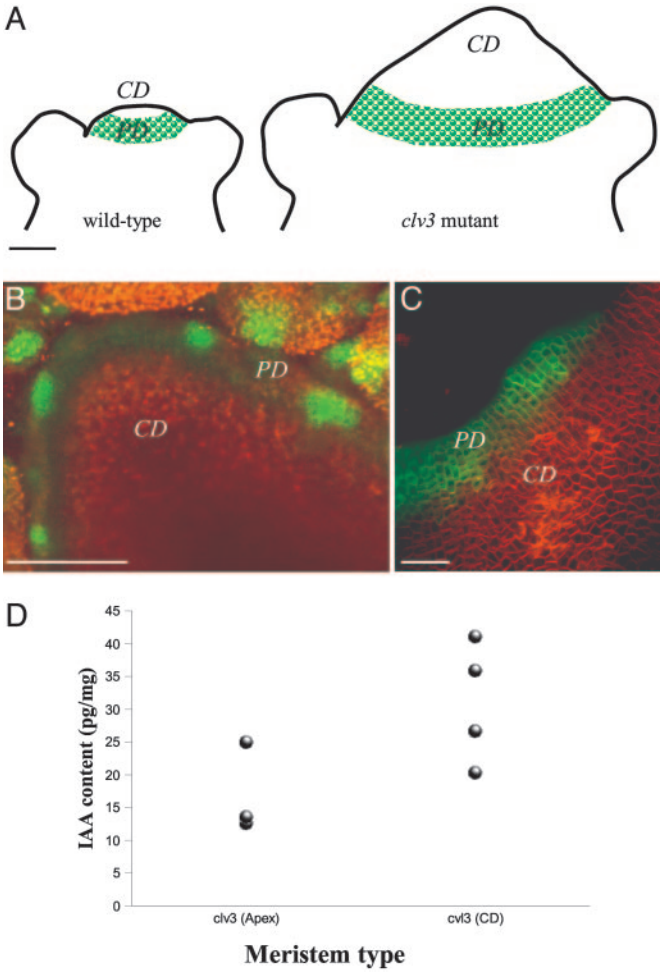


Fig. 6. Quantification of IAA in the central part of the *clv3* meristems. (*A*) Schematic descriptions of wild-type and *clv3* meristems illustrating the enlarged central zone in *clv3* (CD, central domain). The green area represents the periphery domain (PD) where *pDR5*::*GFP* can be expressed. (*B* and *C*) Pattern of *pDR5*::*GFP* expression in *clv3* meristems. (*B*) Global view of a full projection showing that *pDR5* activity is limited to the meristem periphery, with several maxima where the next primordia will be formed. (*C*) Detail of a meristem. (Bars in *A*–*C*, 50 μm.) (*D*) Results of IAA quantification with GCMS in *clv3* meristems. Samples included the young apex (CD + PD + young primordia) or the CD only. For each class, the quantification was performed on four different samples (four dots), each sample containing several meristems. The quantification shows that the central domain of *clv3* meristems concentrates significantly (at 1%) more IAA than the overall apex.

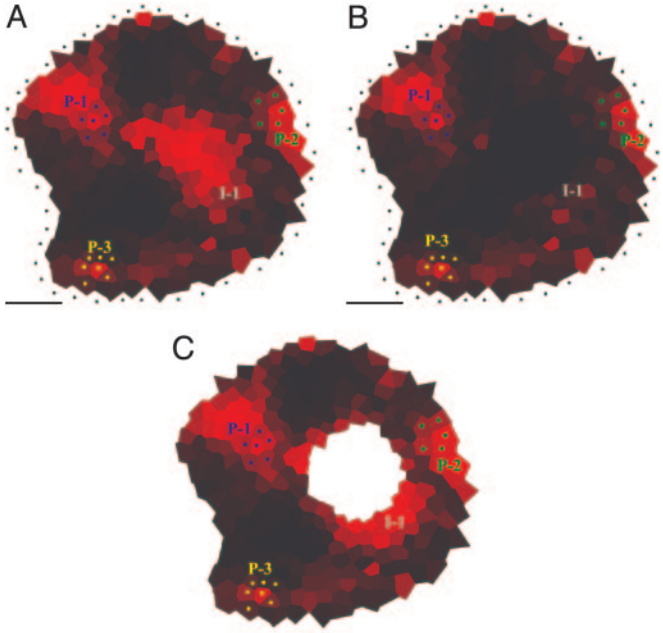


Fig. 7. Testing the importance of auxin accumulation at the meristem summit. (*A*) Simulation of auxin distribution using the standard parameter set (i.e., there are no special instructions for the meristem summit, and auxin is evacuated only via the primordia P-1, P-2, and P-3). (*B*) Simulation of auxin distribution in the same meristem, but this time the auxin arriving at the summit is immediately degraded. As a result, the maximum at the initium I-1 has disappeared. (*C*) Simulation of auxin distribution in the same meristem, but this time, the meristem summit was removed. We defined this summit using the auxin accumulation zone. The initium I-1 is still present.

the auxin-insensitive part of the meristem corresponded to the domain that is under control of the CLV3 pathway. This domain is believed to be equivalent to the so-called "central zone" required for meristem maintenance (23) To determine whether the *clv3* summit contained auxin or not, we next performed GCMS. For this purpose we measured the auxin contents in apices containing the SAM and young flower buds of *clv3* mutants. In addition, samples containing only cells coming from the enlarged meristematic summit of the *clv3* mutant were taken. The results (Fig. 6*D*) showed that the samples enriched in central zone cells contained active IAA and were even enriched in hormone. Thus, the hypothesis that the central domain of the meristem is insensitive to auxin, but contains free IAA, as suggested by the computer simulations and the auxin immunolabeling, was further confirmed by using the IAA quantification in the *clv3* mutant. Several lines of evidence suggest that PIN1 is auxin inducible (24), which might seem in contradiction with our observation that PIN is expressed in the auxin-insensitive center of the meristem. There are two possible explanations for this apparent contradiction. First, PIN expression also might depend on other parameters than auxin, and, second, the meristem summit could be partially sensitive to auxin, via a pathway that does not involve the auxin-responsive elements present in DR5.

Further Simulation to Test the Role of Auxin at the Summit. What could be the function of IAA in the central domain of the meristem? To address this question, we performed additional simulations. These simulations were based on the same rules as before, but in addition the model was instructed to degrade auxin at the meristem summit. In all meristems tested, this additional instruction not only removed the auxin maximum from the meristem summit but also the maximum at the level of the I-1 initium (Fig. 7 *A* and *B*). By contrast, the maxima around the formed primordia were maintained. The results, therefore, suggest that the meristem summit plays an essential role in the creation of novel auxin maxima at the site of the organ primordium founder cells.

Discussion

Together, the simulations and subsequent experiments lead to a model in which auxin coming from the periphery is transported into the central zone of the meristem, which is insensitive to the organ-promoting effect of the hormone. At a certain level of accumulation, auxin can no longer freely enter the meristem summit, and because new auxin is arriving constantly, the hormone will accumulate at the site where the fluxes toward the summit are the most abundant. In a way, this scenario would be analogous to a "traffic jam" at the entry of the meristem. Our

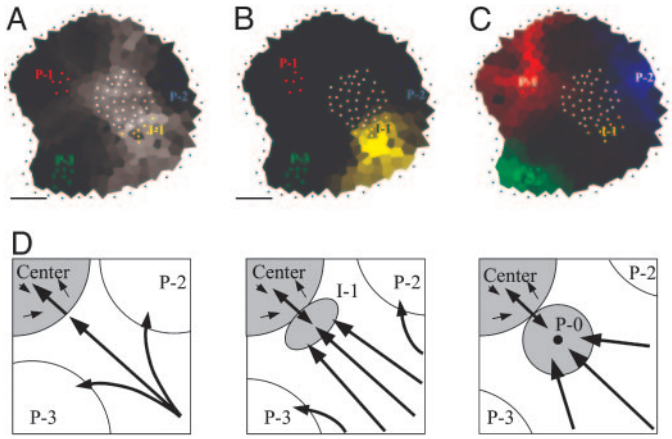


Fig. 8. Auxin fluxes and primordium initiation. (*A–C*) Auxin pathways inferred from a simulation (see *Supporting Text*). The color intensity in each cell is proportional to the contribution of this cell to auxin accumulation in the chosen zone (black: no contribution). The different zones are indicated as groups of colored dots. (*A*) Auxin reaches the summit (gray dots) via corridors between primordia. The most important flux is between P-2 and P-3. I-1 is located at the limit of the summit and the most important flux toward the summit. (*B*) The initium I-1 (yellow dots) is mainly filled by auxin coming from the periphery. PIN patterns suggest that the center contributes little. (*C*) All three primordia receive auxin from the periphery. P-1 (red dots) and P-2 (blue dots) also receive some auxin from the center in contrast to P-3 (green dots). (*D*) Model for the formation of an auxin maximum preceding creation of a primordium. As the distance between P-2 and P-3 increases, more auxin arrives at the meristem center in this sector. Because the center can only absorb a limited amount of auxin, this situation will lead to the formation of an auxin maximum (I-1). Eventually, this maximum will be transformed into a primordium (P-0) where the provascular system behaves as an auxin sink (black dot at the center of the primordium). (Bars, 20 μm.)

simulations predict that this site corresponds precisely to the I-1 area, i.e., the zone where the interprimordium distance is the largest (Fig. 8). At this stage, we have only considered the spiralled phyllotactic patterns observed in *Arabidopsis*. It will certainly be of interest to test our hypothesis that the model is also compatible with other types of phyllotaxis. For this purpose, more extensive simulation efforts using dynamic models will be required.

The results might seem in contradiction with elegant experiments where the tomato meristem summit was ablated by using a laser (15). In this case, no modification in organ positioning was observed, at least for a period of up to four to five plastochrones, suggesting that the meristem center did not play an important role in organ positioning. To clarify this issue, we performed additional simulations, where all cells from the meristem center were removed (Fig. 7*C*). Interestingly, this extra hypothesis did not have an effect on the accumulation of auxin at I-1 in the model. In this context, it should be noted that an ablated meristem center is analogous to a center that no longer accepts auxin. As a consequence, it also would cause an accumulation of auxin at the site where the fluxes are most abundant. Our results are, therefore, fully compatible with the experimental evidence and provide an alternative explanation.

In this study, we have considered the molecular mechanism of auxin flux as a black box, which simply results in a net flux from cell to cell. Hereby, we assume that the PIN-labeled membranes indicate the direction of active transport. Although we show that this approach can lead to testable hypotheses, it might be of interest to include certain processes or parameters that have remained inaccessible for our simulations. For example, it could be useful to consider chemical parameters such as pH-dependent effects that influence permeability of auxin or to include more precise information on auxin concentrations. For this purpose, it will be essential to develop the biological, mathematical, and computer tools required to obtain and analyze quantitative information on these parameters.

In conclusion, our results reveal a robust network of cell interactions that is sufficient to generate auxin distribution patterns consistent with the observed organ positions (25). In addition, they suggest a role for the meristem summit in organ positioning. The next, challenging step will be to understand how the PIN1 proteins themselves are oriented. In this context, two major hypotheses have been proposed. In the first one, the patterns of cell polarity are due to the organization of local gradients of auxin concentrations. This hypothesis was originally used in ref. 12 for designing a computational model of leaf venation formation and was used recently to model various types of leaf venation patterns (26). The phyllotaxis model developed by Jönsson *et al.* (14) is based on a similar hypothesis. In the second hypothesis, the orientation of PIN1 pumps results from a biochemical interpretation of mechanical stresses in the meristem surface. Such a mechanism would provide a possible molecular foundation for mechanical-based models (27, 28). By any means, it will not only be important to identify cellular mechanisms leading to polar localization of PIN1, but we also need to understand how these mechanisms are coordinated at the level of the whole meristem.

Materials and Methods

Immunolabeling of PIN1 Protein. After embedding, the meristems were sectioned perpendicular to the main stems with a thickness of 12–15 μm. After labeling with anti-PIN1, the physical sections were viewed in the confocal microscope to obtain an optimal image of the labeling patterns. In some cases, a single physical section was sufficient to cover the entire dome of the meristem. In other cases, the patterns of two successive sections were combined to cover the dome.

Anti-PIN1. Based on the sequence of *AtPIN1* (gene At1g73590), one potentially antigenic peptide sequence (GPTPRPSNY-EEDGGPA) was selected in the large intracytosolic loop domain of AtPIN1 and used to produce Abs (made by Eurogentec, Seraing, Belgium). This Ab recognizes PIN1, because no labeling is seen at the surface of the meristem in the *pin1* mutant. More detailed characterization of the Ab will be presented elsewhere. After immunostaining, the sections were viewed in a Leica confocal microscope to guarantee an optimal representation of the labeling patterns.

GCMS. For GCMS, the plant tissue was collected in a 1.5-ml microcentrifuge tube and immediately frozen in liquid nitrogen. Then, 0.5 ml of cold 0.05 M phosphate buffer (pH 7.0) containing 0.02% sodium diethyldithiocarbamic acid (antioxidant) was added to the tube, together with $^{13}C_6$-IAA (Cambridge Isotope Laboratories, Cambridge, MA) internal standard (50 pg/mg tissue) and a 3-mm tungsten–carbide bead. The sample was homogenized at 30 Hz in a vibration mill (MM 301, Retsch, Haan, Germany) for 3 min and then extracted under continuous shaking for 15 min at +4°C. After extraction, the pH was adjusted to 2.7 with 1 M HCl. Purification was performed by using solid-phase extraction on a 50-mg BondElut-C18 column (Varian). The column was conditioned with 1 ml of methanol, followed by 1 ml of 1% acetic acid. After application of the sample, the column was washed with 1 ml of 10% methanol in 1% acetic acid. The column was eluted with 1 ml of methanol, and the sample then was evaporated to dryness. Then, 0.2 ml of 2-propanol and 0.5 ml of dichloromethane was added to the sample, followed by 5 μl of 2 M trimethylsilyl-diazomethane in hexane (Sigma-Aldrich). The sample was incubated in room temperature for 30 min, and excess diazomethane then was

destroyed by adding 5 μl of 2 M acetic acid in hexane. After evaporation to dryness, the sample was trimethylsilylated and analyzed by selected-reaction-monitoring GCMS as described.

Modeling Tools. To interpret the labeling patterns in terms of putative auxin distribution, we developed a method relying on the simulation of auxin fluxes on digitized meristems.

Briefly, the method involves the following steps (Fig. 3 *A–G*). First, the membranes of the individual cells are identified on the images of immunolabeled sections. This information is used to reconstruct a graph where the nodes represent the cells and every cell is connected to its neighbors. These connections are used to simulate auxin diffusion from cell to cell. A second type of connections is used to simulate active auxin transport. For this purpose, the cells also are connected via the membranes carrying PIN1 labeling. The latter connections are oriented (represented as arrows in Fig. 3 *D* and *E*) to take into account the direction of PIN1-mediated efflux. By using these maps of interconnected cells, we simulated auxin transport by applying a set of rules based on observations and hypotheses mostly taken from the literature (for a detailed description see *Supporting Text*).

To test the robustness of the auxin distribution patterns, we performed a range of tests in which only one parameter was modified at the time (specified in *Supporting Text*). For each test, the nonvarying parameters were set to values intermediate between those having extreme effects on the simulation (see Tables 1 and 2). The results showed that the patterns were qualitatively insensitive to major changes in diffusion and transport rates. At constant transport strength, the results were qualitatively equivalent for a 13-fold increase in diffusion rates. Conversely, at constant diffusion rate, the results were qualitatively equivalent for a 5-fold increase in transport strength. The patterns, therefore, should be considered as robust.

In a minority of the cells, the immunolabeling was not clear enough to assert the polarity or even the presence of the PIN-protein. Therefore, we classified the different connections into four categories with decreasing confidence level: (*i*) strong signal, (*ii*) strong but unpolarized signal, (*iii*) weak but polarized signal, and (*iv*) weak and unpolarized signal. We next performed the simulations removing the connections *ii–iv*. Because the resulting patterns were not significantly different, we only considered the labeled membranes with the highest confidence level (for details, see *Supporting Text*).

An aspect that was not taken into account was the relative level of immunolabeling. Because there is no experimental evidence of how this level translates into transport rates, we restricted ourselves to recording only the presence/absence of PIN1 on cell walls.

Simulations showed that even an increase of transport rates by a factor of 10 in the young primordia (where PIN1 labeling was the strongest) did not significantly change the final outcome of the simulations (see Fig. 4).

We thank Przemyslaw Prusinkiewicz and Cris Kuhlemeier for critical reading of the text. We also thank Jean-Louis Giavitto (University of Evry, Evry, France) and Olivier Michel (University of Evry, Evry, France) for providing the MGS language. We thank Jiri Friml (University of Tübingen, Tübingen, Germany) for providing the *DR5*::*GFP* line. P.B.d.R. was supported by a fellowship provided by Institut National de la Recherche Agronomique. J.T. was supported by an Action Concertée Incitative (ACI) from the French Ministry of Research and by the Marie Curie program of the European Union. I.B.-C. was supported by the French Ministry of Research.

1. Vernoux, T., Kronenberger, J., Grandjean, O., Laufs, P. & Traas, J. (2000) *Development (Cambridge, U.K.)* **127,** 5157–5165.
2. Reinhardt, D., Pesce, E.-R., Stieger, P., Mandel, T., Baltensperger, K., Bennett, M., Traas, J., Friml, J. & Kuhlemeier, C. (2003) *Nature* **426,** 255–260.
3. Benkova, E., Michniewicz, M., Sauer, M., Teichmann, T., Seifertova, D., Jurgens, G. & Friml, J. (2003) *Cell* **115,** 591–602.
4. Bennett, M. J., Marchant, A., Green, H. G., May, S. T., Ward, S. P., Millner, P. A., Walker, A. R., Schulz, B. & Feldmann, K. A. (1996) *Science* **273,** 948–950.
5. Kramer, E. M. (2004) *Trends Plant Sci.* **9,** 578–582.
6. Vitha, S., Baluska, F., Mews, M. & Volkmann, D. (1997) *J. Histochem. Cytochem.* **45,** 89–95.
7. Traas, J. & Bohn Courseau, I. (2005) *Curr. Opin. Plant Biol.* **8,** 587–592.
8. Vogler, H. & Kuhlemeier, C. (2003) *Curr. Opin. Plant Biol.* **6,** 51–56.
9. Galweiler, L., Guan, C., Müller, A., Wisman, E., Mendgen, K., Yephremov, A. & Palme, K. (1998) *Science* **282**, 2226–2230.
10. Reinhardt, D., Mandel, T. & Kuhlemeier, C. (2000) *Plant Cell* **12,** 507–518.
11. Carraro, N., Peaucelle, A., Laufs, P. & Traas, J. (2006) *Plant Mol. Biol.*, in press.
12. Mitchison, G. J. (1980) *Proc. R. Soc. London Ser. B* **207,** 79–109.
13. Goldsmith, M. H. M., Goldsmith, T. H. & Martin, M. H. (1981) *Proc. Natl. Acad. Sci. USA* **78,** 976–980.
14. Jönsson, H., Heisler, M. G., Shapiro, B. E., Meyerowitz, E. M. & Mjolsness, E. (2006) *Proc. Natl. Acad. Sci. USA* **103,** 1633–1638.
15. Reinhardt, D., Frenz, M., Mandel, T. & Kuhlemeier, C. (2003) *Development (Cambridge, U.K.)* **130,** 4073–4083.
16. Thomas, C., Bronner, R., Molinier, J., Prinsen, E., van Onckelen, H. & Hahne, G. (2002) *Planta* **215,** 577–583.
17. Moctezuma, E. & Lewis, J. F. (1999) *Planta* **209,** 180–186.
18. Grandjean, O., Vernoux, T., Laufs, P., Belcram, K., Mizukami, Y. & Traas, J. (2004) *Plant Cell* **16,** 74–87.
19. Avsian-Kretchmer, O., Cheng, J.-C., Chen, L., Moctezuma, E. & Sung, Z. R. (2002) *Plant Physiol.* **130**, 199–209.
20. Ljung, K., Rishikesh, P., Bhalerao, R. P. & Sandberg, G. (2001) *Plant J.* **28**, 465–474.
21. Edlund, A., Eklöf, S., Sundberg, B., Moritz, T. & Sandberg, G. (1995) *Plant Physiol.* **108,** 1043–1047.
22. Clark, S. E. (2001) *Nat. Rev. Mol. Cell. Biol.* **2,** 276–284.
23. Laux, T. (2003) *Cell* **113,** 281–283.
24. Vieten, A., Vanneste, S., Wisniewska, J., Benková, E., Benjamins, R., Beeckman, T., Luschnig, C. & Friml, J. (2005) *Development (Cambridge, U.K.)* **132,** 4521–4531.
25. Lyndon, R. (1998) *The Shoot Apical Meristem* (Cambridge Univ. Press, Cambridge, U.K.).
26. Runions, A., Fuhrer, M., Lane, B., Federl, P., Rolland-Lagan, A.-G. & Prusinkiewicz, P. (2005) *ACM Trans. Graph.* **24,** 702–711.
27. Green, P., Steel, C. & Rennich, S. (1998) in *Symmetry in Plants*, eds. Jean, R. & Barabé, R. (World Scientific, Toh Tuck Link, Singapore), pp. 359–392.
28. Shipman, P. D. & Newell, A. C. (2005) *J. Theor. Biol.* **236,** 154–197.

L'affluence des résultats de la biologie "à grande échelle" permet d'envisager une nouvelle approche de la biologie du développement fondée sur une modélisation à l'échelle cellulaire. Aujourd'hui, l'organisation génétique et les mécanismes d'action de ces gènes sur les processus de morphogénèse commencent à être analysables. Dans ce contexte, il devient envisageable de comprendre comment l'architecture des plantes peut être modulée par ces mécanismes aux échelles moléculaire et cellulaire.

Dans une première partie de ce travail, des outils de digitalisation et d'analyse de tissus cellulaires et de leur évolution dans le temps ont été développés. Dans ce cadre, nous avons mis en place une chaîne de traitement, partant d'images issues d'un protocole d'observation basé sur la microscopie confocale et allant jusqu'à l'analyse quantitative de caractères morphologiques et leur évolution au cours du temps.

Ensuite, nous avons créé un premier modèle de la croissance du méristème apical caulinaire (une population de cellules souches chez les plantes) dans le but de comprendre le déterminisme du positionnement des organes latéraux de la plante. Pour rendre compte des connaissances biologiques existante, notre modèle a été développé à l'échelle de l'organe et inclus la simulation pour chaque cellule de son état génétique, de son état physiologique et de sa croissance, et la simulation des flux d'hormones entre les cellules. Ce modèle nous a permis de prédire et d'analyser les accumulations d'hormones dans des méristèmes digitalisés. Ces études nous ont conduits à développer un modèle dynamique de fonctionnement du méristème capable de faire émerger des motifs phyllotaxiques sur la base d'une croissance cellulaire et d'une interaction cellule à cellule.

Tous les outils informatiques développés au cours de cette thèse ont été intégrés dans une plate-forme logicielle multi-langages. Cette plate-forme a notamment permis de tester des techniques de développement liés à l'intégration de langages compilés et de langages interprétés dans le cadre de l'étude de la morphogénèse.

Toward a dynamic model of the shoot apical meristem of *Arabidopsis thaliana*.

The huge quantity of data generated by the "large scale" biology allows for a new approach of developmental biology : an approach based on modeling at a cellular level. Nowadays, genetic patterns and genetics pathways involved in morphogenesis begin to be analyzable. In that context, it becomes possible to understand how the plants architecture may be influenced by these cellular or molecular scale mechanisms.

In a first part of this work, tools dedicated to digitization and analysis of cellular tissues and their evolution through time were developed. We set up a software chain for quantitative analysis of morphological characteristics of plant tissues through time, starting from images created using an acquisition protocol based on confocal microscopy.

Then, we developed a first growth model of the shoot apical meristem (a population of stem cells in the plants) to study the initiation of lateral organs. To account for existing biological knowledge, our model was developed at organ scale. The genetic state, the physiological state and the growth of each cell are simulated together with hormones fluxes between cells. This model allowed for prediction and analysis of accumulation zones of hormones in digitized meristems. Thanks to these studies, we were able to develop a dynamic model of the meristem from which phyllotactic patterns emerge from cell growth and cell-cell interactions.

All the computer tools developed during this PhD have been integrated into a multi-language software platform. In particular, this platform was used to test different software development techniques involved in the integration of compiled and interpreted languages in the context of morphogenesis study.

Mots-clefs : auxine, modèle de transport, systèmes dynamiques, reconstruction 3D, imagerie, plate-forme logicielle, lignation cellulaire, phyllotaxie

---

Discipline : Informatique

Laboratoire : UMR AMAP botAnique et bioinforMatique de l'Architecture des Plantes
TA 40 / PS 2 - boulevard de la lironde - 34398 Montpellier - France